Junos® OS

EVPN Feature Guide

Modified: 2018-09-28

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

Junos® OS EVPN Feature Guide

END USER LICENSE AGREEMENT

Table of Contents

List of Figures

List of Tables

About the Documentation

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at https://www.juniper.net/documentation/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at https://www.juniper.net/books.

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   system {
     scripts {
       commit {
         file ex-script.xsl;
       }
     }
   }
   interfaces {
     fxp0 {
       disable;
       unit 0 {
         family inet {
           address 10.0.0.1/24;
         }
       }
     }
   }
   ```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

   ```
   [edit]
   user@host# load merge /var/tmp/ex-script.conf
   load complete
   ```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   commit {
     file ex-script-snippet.xsl; }
   ```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see CLI Explorer.

Documentation Conventions

Table 1 on page xxiv defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Table 2 on page xxiv defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the **configure** command: `user@host> configure`
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host> show chassis alarms` `No alarms currently active`

Table 2: Text and Syntax Conventions (continued)

Convention	Description	Examples
Italic text like this	• Introduces or emphasizes important new terms. • Identifies guide names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS CLI User Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Encloses optional keywords or variables.	**stub <default-metric** ***metric*****>;**
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** **(*string1* | *string2* | *string3*)**
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	**community name members [** ***community-ids*** **]**

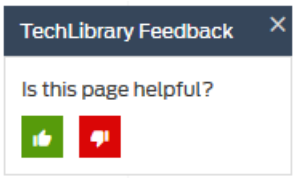
Indention and braces ({ })	Identifies a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback so that we can improve our documentation. You can use either of the following methods:

- Online feedback system—Click TechLibrary Feedback, on the lower right of any page on the Juniper Networks TechLibrary site, and do one of the following:



  - Click the thumbs-up icon if the information on the page was helpful to you.
  - Click the thumbs-down icon if the information on the page was not helpful to you or if you have suggestions for improvement, and use the pop-up form to provide feedback.

- E-mail—Send your comments to techpubs-comments@juniper.net. Include the document or topic name, URL or page number, and software version (if applicable).

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or Partner Support Service support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at https://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit https://www.juniper.net/support/warranty/.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: https://www.juniper.net/customers/support/
- Search for known bugs: https://prsearch.juniper.net/
- Find product documentation: https://www.juniper.net/documentation/

- Find solutions and answer questions using our Knowledge Base: https://kb.juniper.net/
- Download the latest versions of software and review release notes: https://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: https://kb.juniper.net/InfoCenter/
- Join and participate in the Juniper Networks Community Forum: https://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: https://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://entitlementsearch.juniper.net/entitlementsearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at https://www.juniper.net/cm/.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see https://www.juniper.net/support/requesting-support.html.

PART 1

Features Common to EVPN-VXLAN and EVPN-MPLS

CHAPTER 1

Configuring Routing Instances for EVPN

Configuring EVPN Routing Instances

To configure an EVPN routing instance, complete the following configuration on the PE router (or on the MPLS edge switch or QFX Series switch) within the EVPN service provider's network:

1. Configure the EVPN routing instance name using the **routing-instances** statement at the **[edit]** hierarchy level:

   routing-instances *routing-instance-name* {...}

2. Configure the **evpn** option for the **instance-type** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

   instance-type evpn;

NOTE: For MX Series devices, EX Series, and QFX Series switches, you can include multiple IFLs of an Ethernet segment identifier (ESI) across different bridge-domains or VLANs of an EVPN routing instance in all-active mode. However, you cannot include multiple IFLs of the same ESI within the same bridge-domain or VLAN.

3. Configure the interfaces for handling EVPN traffic between the MES or PEs and the CE device using the **interface** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

   ```
   interface interface-name;
   ```

4. Configure a VLAN identifier for the EVPN routing instance using the **vlan-id** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

   NOTE: For QFX Series, set the VLAN ID to none.

   ```
   vlan-id (vlan-id | all | none);
   ```

5. Configure a route distinguisher on a PE router by including the **route-distinguisher** statement:

   ```
   route-distinguisher (as-number:number | ip-address:number);
   ```

   Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN routing instances need a route distinguisher to help BGP to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs. If you configure different VPN routing instances with the same route distinguisher, the commit fails.

   For a list of the hierarchy levels at which you can include this statement, see the statement summary for this statement.

   The route distinguisher is a 6-byte value that you can specify in one of the following formats:

   - ***as-number*:*number***, where ***as-number*** is an autonomous system (AS) number (a 2-byte value) and ***number*** is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the Internet service provider's (ISP's) own or the customer's own AS number.

     NOTE: The automatic derivation of the BGP route target (auto-RT) for advertised prefixes is supported on a 2-byte AS number only.

   - ***ip-address*:*number***, where ***ip-address*** is an IP address (a 4-byte value) and ***number*** is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the **router-id** statement, which is a nonprivate address in your assigned prefix range.

6. Configure either import and export policies for the EVPN routing table, or configure the default policies using the **vrf-target** statement configured at the **[edit routing-instances *routing-instance-name*]** hierarchy level.

See *Configuring Policies for the VRF Table on PE Routers in VPNs*.

7. Configure each EVPN interface for the EVPN routing instance:

   a. Configure each interface using the interface statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

   b. Configure interface encapsulation for the CE facing interfaces at the **[edit interfaces *interface-name* encapsulation]** hierarchy level. Supported encapsulations, except for EX9200 switches and QFX Series switches, are: **(ethernet-bridge | vlan-bridge | extended-vlan-bridge)**. Supported encapsulations for EX9200 switches are: **(extended-vlan-bridge | flexible-ethernet-services)**. Supported encapsulation for QFX Series switches is **vxlan**.

   c. (Optional) Allow the EVPN to establish a connection to the CE device even if the CE device interface encapsulation and the EVPN interface encapsulations do not match by including the *ignore-encapsulation-mismatch* statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level.

   d. (Optional) (Not available on EX9200 switches) Specify a static MAC address for a logical interface in a bridge domain using the *static-mac* statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level.

8. Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN routing instance by including the interface-mac-limit statement.

   You can configure the same limit for all interfaces configured for a routing instance by including this statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level. You can also configure a limit for a specific interface by including this statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level.

   By default, packets with new source MAC addresses are forwarded after the MAC address limit is reached. You can alter this behavior by including the **packet-action drop** statement at either the **[edit routing-instances *routing-instance-name* protocols evpn interface-mac-limit]** or the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level. If you configure this statement, packets from new source MAC addresses are dropped once the configured MAC address limit is reached.

9. Specify the MPLS label allocation setting for the EVPN by including the *label-allocation* statement with the **per-instance** option at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

   If you configure this statement, one MPLS label is allocated for the specified EVPN routing instance.

10. Enable MAC accounting for the EVPN by including the *mac-statistics* statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

11. Specify the number of addresses that can be stored in the MAC routing table using the *mac-table-size* statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

    You can optionally configure the *packet-action* **drop** option to specify that packets for new source MAC addresses be dropped once the MAC address limit is reached. If you do not configure this option, packets for new source MAC addresses are forwarded.

12. Disable MAC learning by including the *no-mac-learning* statement at either the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level to apply this behavior to all of the devices configured for an EVPN routing instance or at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level to apply this behavior to just one of the CE devices.

Related Documentation

- *Configuring Policies for the VRF Table on PE Routers in VPNs*
- *Configuring Routing Instances on PE Routers in VPNs*
- Tracing EVPN Traffic and Operations on page 19

Configuring EVPN Routing Instances on EX9200 Switches

To configure an EVPN routing instance, complete the following configuration on the PE router (or on the MPLS edge switch) within the EVPN service provider's network:

1. Configure the EVPN routing instance name using the **routing-instances** statement at the **[edit]** hierarchy level:

   routing-instances *routing-instance-name* {...}

2. Configure the **evpn** option for the **instance-type** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

   instance-type evpn;

3. Configure the interfaces for handling EVPN traffic between the MES and the CE device using the **interface** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

   interface *interface-name*;

4. Configure a VLAN identifier for the EVPN routing instance using the **vlan-id** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

   vlan-id (*vlan-id* | all | none);

5. Configure a route distinguisher on a PE router by including the **route-distinguisher** statement:

   ```
   route-distinguisher (as-number:number | ip-address:number);
   ```

   Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN routing instances need a route distinguisher to help BGP to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs. If you configure different VPN routing instances with the same route distinguisher, the commit fails.

   For a list of the hierarchy levels at which you can include this statement, see the statement summary for this statement.

   The route distinguisher is a 6-byte value that you can specify in one of the following formats:

   - ***as-number*:*number***, where ***as-number*** is an autonomous system (AS) number (a 2-byte value) and ***number*** is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the Internet service provider's (ISP's) own or the customer's own AS number.
   - ***ip-address*:*number***, where ***ip-address*** is an IP address (a 4-byte value) and ***number*** is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the **router-id** statement, which is a nonprivate address in your assigned prefix range.

6. Configure either import and export policies for the EVPN routing table, or configure the default policies using the **vrf-target** statement configured at the **[edit routing-instances *routing-instance-name*]** hierarchy level.

   See *Configuring Policies for the VRF Table on PE Routers in VPNs*.

7. Configure each EVPN interface for the EVPN routing instance:

   a. Configure interface encapsulation for the CE facing interfaces at the **[edit interfaces *interface-name* encapsulation]** hierarchy level. . Supported encapsulations for EX9200 switches are: **(extended-vlan-bridge | flexible-ethernet-services | vlan-bridge)**.

   b. Configure **vlan-bridge** encapsulation on the logical interface at the **[edit interfaces *interface-name* flexible-vlan-tagging encapsulation flexible-ethernet-services unit 0 encapsulation]** hierarchy level.

   c. (Optional) Allow the EVPN to establish a connection to the CE device even if the CE device interface encapsulation and the EVPN interface encapsulations do not match by including the *ignore-encapsulation-mismatch* statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level.

8. Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN routing instance by including the interface-mac-limit statement.

You can configure the same limit for all interfaces configured for a routing instance by including this statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level. You can also configure a limit for a specific interface by including this statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level.

By default, packets with new source MAC addresses are forwarded after the MAC address limit is reached. You can alter this behavior by including the **packet-action drop** statement at either the **[edit routing-instances *routing-instance-name* protocols evpn interface-mac-limit]** or the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level. If you configure this statement, packets from new source MAC addresses are dropped once the configured MAC address limit is reached.

9. Enable MAC accounting for the EVPN by including the *mac-statistics* statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

10. Specify the number of addresses that can be stored in the MAC routing table using the *mac-table-size* statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

    You can optionally configure the *packet-action* **drop** option to specify that packets for new source MAC addresses be dropped once the MAC address limit is reached. If you do not configure this option, packets for new source MAC addresses are forwarded.

11. Disable MAC learning by including the *no-mac-learning* statement at either the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level to apply this behavior to all of the devices configured for an EVPN routing instance or at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level to apply this behavior to just one of the CE devices.

Related Documentation

- *Configuring Policies for the VRF Table on PE Routers in VPNs*
- *Configuring Routing Instances on PE Routers in VPNs*
- Tracing EVPN Traffic and Operations on page 19

Overview of MAC Mobility

MAC mobility describes the scenario where a host moves from one Ethernet segment to another segment in the EVPN network. Provider Edge (PE) devices discover the host MAC address from its local interfaces or from remote PE devices. When a PE device learns of a new local MAC address, it sends a MAC advertisement route message to other devices in the network. During this time, there are two advertised routes and the PE devices in the EVPN network must decide which of the MAC advertisement messages to use.

To determine the correct MAC address location, PE devices use the MAC mobility extended community field, as defined in RFC 7432, in the MAC advertisement route message. The MAC mobility extended community includes a static flag and a sequence number. The static flag identifies pinned MAC addresses that should not be relocated. The sequence number identifies newer MAC advertisement messages. Starting at 0, the sequence number is incremented for every MAC address mobility event. PE devices running Junos OS apply the following precedence order in determining the MAC advertisement route to use:

1. Advertisement routes with a local pinned MAC address (static MAC address).
2. Advertisement routes with a remote pinned MAC address (static MAC address).
3. Advertisement routes with a higher sequence number.

NOTE: When there are two advertisement route messages for pinned MAC addresses with different routes or two advertisement route messages with the same sequence number, the local device chooses the advertisement route message from the PE device with the lower IP address.

Figure 1 on page 9 illustrates a network where a MAC address is relocated from PE1 to PE2. Before the move, a MAC advertisement route message sent by PE1 has the active route for all PE devices in the network. After the relocation, PE2 learns of the new local MAC address and sends an updated MAC advertisement route message.
Table 3 on page 10 lists the action taken by each PE device based on the two MAC advertisements. The PE device generates a syslog message when it encounters conflicts with a pinned MAC address.

NOTE: Table 3 on page 10 includes use cases with pinned MAC addresses. These use cases do not apply to PE devices that do not support MAC pinning. To determine whether or not MAC pinning is supported by a particular Juniper Networks device or Junos OS release, see Feature Explorer.

Figure 1: MAC Mobility in an EVPN Network

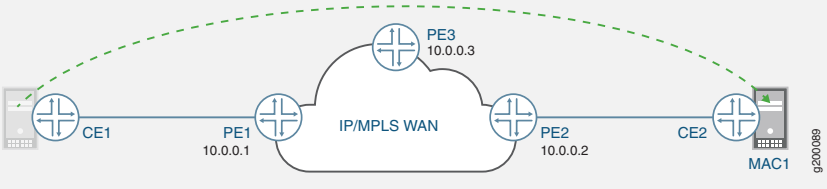


Table 3: MAC Advertisement Routes on PE Devices

MAC Advertisement	PE1	PE2	PE3
PE1: MAC address with a sequence number (n). PE2: MAC address with the sequence number incremented by one (n+1).	Install the remote MAC advertisement route from PE2 because it has a higher sequence number (n+1).	Advertise the local MAC route because it has a higher sequence number (n+1).	Install the remote MAC advertisement route from PE2 because it has a higher sequence number (n+1).
PE1: MAC address with a sequence number (n). PE2: MAC address with the same sequence number (n).	Advertise the local MAC route because PE1 has the lower IP address (10.0.0.1).	Install the remote MAC advertisement route from PE1 because PE1 has the lower IP address (10.0.0.1).	Use the MAC advertisement route from PE1 because PE1 has the lower IP address (10.0.0.1).
PE1: Pinned MAC address with the static bit set. PE2: MAC address and a sequence number (n).	Advertise the local MAC route because it is a pinned MAC address. Generate a syslog message.	Install the remote MAC advertisement route from PE1 because it is a pinned MAC address.	Use the MAC advertisement route from PE1 because it is a pinned MAC address. Generate a syslog message.
PE1: MAC address with a sequence number (n). PE2: Pinned MAC address with the static bit set.	Install the remote the MAC advertisement route from PE2 because it is a pinned MAC address.	Advertise local MAC route because it is a pinned MAC address. Generate a syslog message.	Install the remote MAC advertisement route from PE2 because it is a pinned MAC address.
PE1: Pinned MAC address with static bit set. PE2: Pinned MAC address with static bit set.	Advertise the local MAC route because it is a local pinned MAC address. Generate a syslog message.	Advertise the local MAC route because it is a local pinned MAC address. Generate a syslog message.	Use the MAC advertisement route from PE1 because PE1 has the lower IP address (10.0.0.1). Generate a syslog message.

Related Documentation

- EVPN MAC Pinning Overview on page 232
- clear evpn duplicate-mac-suppression on page 1116
- duplicate-mac-detection on page 1051

Changing Duplicate MAC Address Detection Settings

When a host is physically moved or when a host is reconfigured on a different Ethernet segment, the PE device sends an updated MAC advertisement route to other PE devices to update their route table. If there is a misconfiguration in the network, MAC advertisement messages oscillate between the different routes causing MAC address flapping. This makes the network more vulnerable and wastes network resources. By default, Junos OS detects and suppresses duplicate MAC addresses. Optionally, you can also configure the length of time that the duplicate MAC address is suppressed. When the PE device encounters duplicate MAC addresses, Junos OS generates a syslog message.

To change the duplicate MAC address detection settings , include the **duplicate-mac-detection** statement at either the **[edit routing-instances *routing-instance-name* protocols]** hierarchy level or the**[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols]** hierarchy level:

```
evpn
  duplicate-mac-detection {
    detection-threshold detection-threshold;
    detection-window seconds;
    auto-recovery-time minutes;
  }
```

You can modify the following options under the **duplicate-mac-detection** statement:

- **detection-window**—The time interval used in detecting a duplicate MAC address. The value can be from 5 through 600 seconds. The default is 180 seconds
- **detection-threshold**—The number of MAC mobility events that are detected for a given MAC address within the **detection-window** before it is identified as a duplicate MAC address. Once the detection threshold is reached, updates for the MAC address are suppressed. The value can be from 2 through 20. The default is 5.
- **auto-recovery-time**—(Optional) The length of time a device suppresses a duplicate MAC address. At the end of this duration, MAC address updates will resume. The value can be from 5 through 360 minutes. If a value is not specified, then the MAC address continues to be suppressed.

NOTE: **To ensure that the mobility advertisements have sufficient time to age out, set an auto-recovery-time greater than the detection-window.**

To manually clear the suppression of duplicate MAC addresses, use the **clear evpn duplicate-mac-suppression** command.

To view MAC duplicate addresses in the EVPN MAC database, use the **show evpn database** command. The following example displays a sample output. The output fields related

to duplicate MAC detections are State, Mobility history, and MAC advertisement route status:

```
user@PE1> show evpn database mac-address 00:00:00:00:00:02 extensive

Instance: ALPHA

VLAN ID: 100, MAC address: 00:00:00:00:00:02
  State: 0x1 <Duplicate-Detected>
  Mobility history
    Mobility event time     Type    Source                                  Seq
 num
    Aug 03 17:22:28.585619  Local   ge-0/0/2.0                               31
    Aug 03 17:22:30.307198  Remote  10.255.0.3                               32
    Aug 03 17:22:37.611786  Local   ge-0/0/2.0                               33
    Aug 03 17:22:39.289357  Remote  10.255.0.3                               34
    Aug 03 17:22:45.609449  Local   ge-0/0/2.0                               35
  Source: ge-0/0/2.0, Rank: 1, Status: Active
    Mobility sequence number: 35 (minimum origin address 10.255.0.2)
    Timestamp: Aug 03 17:22:44 (0x5983be54)
    State: <Local-MAC-Only Local-To-Remote-Adv-Allowed>
    MAC advertisement route status: Not created (duplicate MAC suppression)
    IP address: 10.0.0.2
  Source: 10.255.0.3, Rank: 2, Status: Inactive
    MAC label: 300176
    Mobility sequence number: 34 (minimum origin address 10.255.0.3)
    Timestamp: Aug 03 17:22:39 (0x5983be4f)
    State: <>
    MAC advertisement route status: Not created (inactive source)
    IP address: 10.0.0.3
```

Related Documentation

- clear evpn duplicate-mac-suppression on page 1116
- duplicate-mac-detection on page 1051

EVPN Type-5 Route with VXLAN encapsulation for EVPN-VXLAN

EVPN is a flexible solution that uses Layer 2 overlays to interconnect multiple edges (virtual machines) within a data center. Traditionally, the data center is built as a flat Layer 2 network with issues such as flooding, limitations in redundancy and provisioning, and high volumes of MAC addresses learned, which cause churn at node failures. EVPN is designed to address these issues without disturbing flat MAC connectivity.

VXLAN is an overlay technology that encapsulates MAC frames into a UDP header at Layer 2. Communication is established between two virtual tunnel endpoints (VTEPs). VTEPs encapsulate the virtual machine traffic into a VXLAN header, as well as strip off the encapsulation. Virtual machines can only communicate with each other when they belong to the same VXLAN segment. A 24-bit virtual network identifier (VNID) uniquely identifies the VXLAN segment. This enables having the same MAC frames across multiple VXLAN segments without traffic crossover. Multicast in VXLAN is implemented as Layer 3 multicast, in which endpoints subscribe to groups.

When a Bridge Domain (BD) is not L2 extended across Data Centers (DC), the IP subnet belonging to the BD is confined within a single DC. If all BDs within each DC network satisfy this requirement, there is no longer a need to advertise MAC+IP route for each tenant between data centers as host routes for the tenants can be aggregated. Thus the L2 inter-DC connectivity issue can be simply transformed to an inter-DC L3 IP prefix reachability issue.

Starting with Junos OS Release 17.1, the EVPN type-5 IP prefix route advertises the IP prefixes between the DCs. Unlike the type-2 EVPN MAC advertisement route, the EVPN type-5 IP prefix route seperates the host MAC address from its IP address and provides a clean advertisement of an IP prefix for the bridge domain.

Junos OS Release 17.1 also supports:

- Inter-DC connectivity with VXLAN encapsulation for EVPN/VXLAN by using the EVPN type 5 IP prefix route. Each BD within a DC is not L2 extended. If EVPN/VXLAN is enabled between DC GW (Data Center Gateway) router and ToR while providing inter-DC connectivity, the spine, which acts as a DC GW router, is capable of performing L3 routing and IRB functions.
- Inter-pod connectivity with VXLAN encapsulation by using the EVPN type-5 IP prefix route. The solution provided does not address the L2 extension problem when a BD is stretched across different pods. The spines that provide the inter-pod connectivity is able to perform L3 routing and IRB functions.

Related Documentation

- EVPN Type-5 Route with MPLS encapsulation for EVPN-MPLS on page 14

EVPN Type-5 Route with MPLS encapsulation for EVPN-MPLS

EVPN is a flexible solution that uses Layer 2 overlays to interconnect multiple edges (virtual machines) within a data center. Traditionally, the data center is built as a flat Layer 2 network with issues such as flooding, limitations in redundancy and provisioning, and high volumes of MAC addresses learned, which cause churn at node failures. EVPN is designed to address these issues without disturbing flat MAC connectivity.

The MPLS infrastructure allows you to take advantage of the MPLS functionality provided by the Junos operating system (Junos OS), including fast reroute, node and link protection, and standby secondary paths.

Starting with Junos OS Release 17.1, to support the interconnection through EVPN type-5 route in the metro service application using EVPN/MPLS, an MPLS tunnel is required instead of a VXLAN tunnel.

NOTE: **L3VPN forwarding based on pure type-5 without overlay next-hop is only supported.**

In the control plane EVPN type-5 is used to advertise IP prefix for inter-subnet connectivity across metro peering points. To reach the end host through the connectivity provided by the EVPN type-5 prefix route, data packets are sent out as an IP packet encapsulated in the MPLS across the metro peering points.

Related Documentation

- EVPN Type-5 Route with VXLAN encapsulation for EVPN-VXLAN on page 13

Understanding EVPN Pure Type-5 Routes

Ethernet VPN (EVPN) offers an end-to-end solution for data center Virtual Extensible LAN (VXLAN) networks. A main application of EVPN is Data Center Interconnect (DCI), which provides the ability to extend Layer 2 connectivity between different data centers. EVPN uses the concept of route types to establish sessions between the provider edge and the customer edge. There are many route types. A type-5 route, also called the IP prefix route, is used to communicate between data centers (DC) when the Layer 2 connection does not extend across DCs and the IP subnet in a Layer 2 domain is confined within a single DC. In this scenario, the type-5 route enables connectivity across DCs by advertising the IP prefixes assigned to the VXLANs confined within a single DC. Data packets are sent as Layer 2 Ethernet frames encapsulated in the VXLAN header. Additionally, the gateway device for the DC must be able to perform Layer 3 routing and provide IRB functionality.

NOTE: **Only pure type-5 routes are supported. Support was added in Junos OS Release 15.1X53-D30 for QFX10002 switches only. Starting with Junos OS Release 15.1X53-D60, pure type-5 routes are also supported on QFX10008 and QFX10016 switches. Starting with Junos OS Release 17.4R1, pure type-5 routes are supported on standalone QFX5110 switches only. Pure type-5 routes are not supported on QFX5110 Virtual Chassis and Virtual Chassis Fabric (VCF).**

A pure type-5 route operates without an overlay next hop or a type-2 route for recursive route resolution. With pure type-5 routing, the type-5 route is advertised with the MAC extended community so that the type-5 route provides all necessary forwarding information required for sending VXLAN packets in the data plane to the egress network virtual endpoint. There is not need to use an IP address as an overlay next hop to interconnect Layer 3 virtual routing and forwarding (VRF) routes sitting in different data centers. Because no type-2 routes are used for route recursive resolution, this provisioning model is also called the IP-VRF-to-IP-VRF model without a core-facing IRB interface.

Defining EVPN-VXLAN Route Types

The EVPN-VXLAN route types are:

- Type-1 route, Ethernet autodiscovery route—Type-1 routes are for networkwide messages. Ethernet autodiscovery routes are advertised on a per end virtual identifier (EVI) and per Ethernet segment identifier (ESI) basis. The Ethernet autodiscovery routes are required when a customer edge (CE) device is multihomed. When a CE device is single-homed, the ESI is zero. This route type is supported by all EVPN switches and routers.

  An ESI can participate in more than one broadcast domain; for example, when a port is trunked. An ingress provider edge (PE) device that reaches the MAC on that ESI must have type-1 routes to perform split horizon and fast withdraw. Therefore, a type-1 route for an ESI must reach all ingress PE devices importing a virtual network identifier (VNI) or tag (broadcast domains) in which that ESI is a member. The Junos OS supports this by exporting a separate route target for the type-1 route.

- Type-2 route, MAC with IP advertisement route—Type-2 routes are per-VLAN routes, so only PEs that are part of a VNI need these routes. EVPN allows an end host's IP and MAC addresses to be advertised within the EVPN Network Layer reachability information (NLRI). This allows for control plane learning of ESI MAC addresses. Because there are many type-2 routes, a separate route-target auto-derived per VNI helps to confine their propagation. This route type is supported by all EVPN switches and routers.

- Type-3 route, inclusive multicast Ethernet tag route—Type-3 routes are per-VLAN routes; therefore, only PE devices that are part of a VNI need these routes. An inclusive multicast Ethernet tag route sets up a path for broadcast, unknown unicast, and multicast (BUM) traffic from a PE device to the remote PE device on a per-VLAN, per-EVI basis. Because there are many type-3 routes, a separate route-target auto-derived per VNI helps in confining their propagation. This route type is supported by all EVPN switches and routers.

- Type-4 route, Ethernet segment Route—An Ethernet segment identifier (ESI) allows a CE device to be multihomed to two or more PE devices—in single/active or active/active mode. PE devices that are connected to the same Ethernet segment discover each other through the ESI. This route type is supported by all EVPN switches and routers.

- Type-5 route, IP prefix Route—An IP prefix route provides encoding for inter-subnet forwarding. In the control plane, EVPN type-5 routes are used to advertise IP prefixes for inter-subnet connectivity across data centers. To reach a tenant using connectivity provided by the EVPN type-5 IP prefix route, data packets are sent as Layer 2 Ethernet frames encapsulated in the VXLAN header over the IP network across the data centers.

- Type-6 route, selective multicast Ethernet tag routes.

- Type-7 route, network layer reachability information (NLRI) to sync IGMP joins.

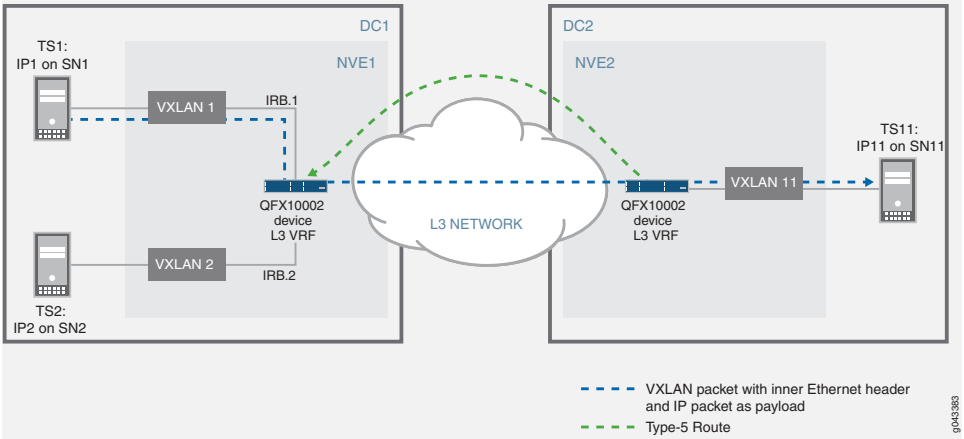
- Type-8 route, NLRI to sync IGMP leaves.

Implementing Pure Type-5 Routes in an EVPN-VXLAN Environment

You can use EVPN pure type-5 routes on QFX10000 switches to communicate between data centers through a Layer 3 network. See Figure 2 on page 17. A unified EVPN control plane accomplishes L3 route advertisement between multiple data center locations so that you do not have to rely on an additional L3 VPN protocol family. On the customer

edge (CE), hosts such as servers, storage devices, or any bare-metal devices are attached to leaf switches on the provider edge. Between those leaf devices, an MP-BGP session is established for EVPN routes to be used in the overlay control protocol. .

Figure 2: EVPN-VXLAN Connection with Pure Type-5 Route Between Two Data Centers



A global unique virtual network identifier (VNI) is provisioned for each customer L3 VRF and identifies the customer L3 VRF at the egress. A chassis MAC is used as the inner destination MAC (DMAC) for the VXLAN packet. The chassis MAC is shared among different customer L3 VRF instances

NOTE: When a virtual machine (VM) moves from one QFX10000 data center to another, a type-5 route no longer works. This is because both the VXLAN and IP subnet that belong to the VM are no longer confined within a single data center.

NOTE: For an example of communicating within a single data center without type-5 routing, see "Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center" on page 302.

Understanding Pure Type 5-Route Forwarding

Pure type-5 route forwarding is also called the IP-VRF-to-IP-VRF (virtual routing and forwarding) model. In IP-based computer networks, Layer 3 VRF allows multiple instances of a routing table to coexist within the same router at the same time. Because the routing instances are independent, the same or overlapping IP addresses can be used without conflicting with each other. In this scenario, for a given tenant, such as an IP VPN service, a network virtualization edge (NVE) has one MAC VRF, which consists of multiple VXLANs (one VXLAN per VLAN). The MAC VRFs on an NVE for a given tenant are associated with an IP VRF corresponding to that tenant (or IP VPN service) through their IRB interfaces.

A global unique VNI is provisioned for each customer Layer 3 VRF. The VNI is used to identify the Layer 3 VRF for the customer on each data center.

Understanding EVPN Pure Type-5 Routes and Local Preferences

On QFX10000 switches running Junos OS Release 15.1X53-D65 or later, the local preference setting for an Ethernet VPN (EVPN) pure type-5 route is inherited by IP routes that are derived from the EVPN type 5 route. Further, when selecting an IP route for incoming traffic, the QFX10000 switches consider the local preference of the route. A benefit of the QFX10000 switches including local preference in their route selection criteria is that you can set up a policy to manipulate the local preference, thereby controlling which route the switch selects.

Advantages of Using EVPN Pure Type-5 Routing

There are two main advantages to using EVPN pure type-5 routing:

- There is no need to exchange all host routes between data center locations. This results in smaller requirements for the routing information base (RIB), also known as the routing table, and the forwarding information base (FIB), also known as the forwarding table, on DCI equipment.
- There is no need to use multiple protocol families, such as both EVPN and an L3 VPN, to advertise L2 and L3 reachability information.

Best Practices and Caveats

BEST PRACTICE: You can use pure type-5 route within a single data center to interconnect points of delivery (pods) as long as the IP prefix can be confined within the pod.

BEST PRACTICE: Note that there are differences between EVPN VXLAN and EVPN MPLS. EVPN VXLAN exports a separate route target for type-1 routes. EVPN-MPLS exports the type-1 route with the collective set of route-targets of the VNI or tags (broadcast domains) in which the Ethernet segment identifier is participating.

NOTE: You cannot use Contrail with pure type-5 route.

Release History Table

Release	Description
17.4R1	Starting with Junos OS Release 17.4R1, pure type-5 routes are supported on standalone QFX5110 switches only.
15.1X53D60	Starting with Junos OS Release 15.1X53-D60, pure type-5 routes are also supported on QFX10008 and QFX10016 switches.
15.1X53-D30	Only pure type-5 routes are supported. Support was added in Junos OS Release 15.1X53-D30 for QFX10002 switches only.

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- ip-prefix-routes on page 1068
- ip-prefix-support on page 1072

Tracing EVPN Traffic and Operations

To configure the EVPN routing instance to trace a variety of different parameters related to EVPN operation:

1. Specify the name of one or more EVPN trace files using the **file** option for the **traceoptions** statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level:

```
traceoptions {
   file filename <files number> <size size> <world-readable | no-world-readable>;
}
```

   The **file** option includes the following sub-options:

   - ***filename***—Specify the name of the file to receive the output of the tracing operation. Enclose the name within quotation marks. All files are placed in the directory **/var/log**.
   - **files *number***—(Optional) Maximum number of trace files. When a trace file named **trace-file** reaches its maximum ***size***, it is renamed **trace-file.0**, then **trace-file.1**, and so on, until the specified maximum ***number*** of trace files specified is reached. Then the oldest trace file is overwritten.
   - **size *size***—(Optional) Maximum size of each trace file. When a trace file named **trace-file** reaches its maximum size, it is renamed **trace-file.0**, then **trace-file.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten.
   - **world-readable | no-world-readable**—(Optional) Enable unrestricted file access or restrict file access to the user who created the file.

2. Specify the **flag** option for the **traceoptions** statement:

```
traceoptions {
  flag flag <flag-modifier> <disable>;
}
```

The **flag** option allows you to specify the scope of the trace by including one of the following sub-options:

- **all**—All EVPN tracing options

  error—Error conditions
- **general**—General events
- **mac-database**—MAC route database in the EVPN routing instance
- **nlri**—EVPN advertisements received or sent by means of the BGP
- **normal**—Normal events
- **oam**—OAM messages
- **policy**—Policy processing
- **route**—Routing information
- **state**—State transitions
- **task**—Routing protocol task processing
- **timer**—Routing protocol timer processing
- **topology**—EVPN topology changes caused by reconfiguration or advertisements received from other PE routers using BGP

You can also specify one of the following modifiers for any of the traceoptions flags:

- **detail**—Provide detailed trace information.
- **disable**—Disable this trace flag.
- **receive**—Trace received packets.
- **send**—Trace sent packets.

Related Documentation

- Configuring EVPN Routing Instances on page 3
- traceoptions on page 1087

Migrating From BGP VPLS to EVPN Overview

For service providers using both BGP VPLS and EVPN networks, there is a need to interconnect these networks. Prior to Junos OS Release 18.1, a logical tunnel interfaces on the interconnection point of the VPLS and EVPN routing instances was used for this purpose. In this case, the PE devices in each network are unaware of the PE devices in the other technology network. Starting in Junos OS Release 18.1, a solution is introduced

for enabling staged migration from BGP VPLS toward EVPN on a site-by-site basis for every VPN routing instance. In this solution, the PE devices running EVPN and VPLS for the same VPN routing instance and single-homed segments can coexist. The solution supports single-active redundancy of multi-homed networks and multi-homed devices for EVPN PEs. With single-active redundancy, the participant VPN instances may span across both EVPN PEs and VPLS PEs as long as single-active redundancy is employed by EVPN PEs.

NOTE: **For migration from BGP VPLS, you should expect some traffic loss when family EVPN is enabled to carry EVPN NLRIs. Routing-instance migration to EVPN should see minimal loss.**

The following sections describe the migration from BGP VPLS to EVPN:

Technology Overview and Benefits

VPLS is an Ethernet-based point-to-multipoint Layer 2 VPN. This technology allows you to connect geographically dispersed data center LANs to each other across an MPLS backbone while maintaining Layer 2 connectivity. The high availability features defined in VPLS standards (such as LER dual homing) and topology autodiscovery features using BGP signaling make VPLS scalable and easy to deploy. Because VPLS uses MPLS as its core, it provides low latency variation and statistically bound low convergence times within the MPLS network.

EVPN, on the other hand, is a combined Layer 2 and Layer 3 VPN solution that is more scalable, resilient, and efficient than current technologies. It provides several benefits including greater network efficiency, reliability, scalability, virtual machine (VM) mobility, and policy control for service providers and enterprises.

Although VPLS is a widely deployed Layer 2 VPN technology, service provider networks migrate to EVPN on account of the scaling benefits and ease of deployment. Some of the benefits of EVPN include:

- Control plane traffic is distributed with BGP and the broadcast and multicast traffic is sent using a shared multicast tree or with ingress replication.
- Control plane learning is used for MAC and IP addresses instead of data plane learning. MAC address learning requires the flooding of unknown unicast and ARP frames; whereas, IP address learning does not require any flooding.
- Route reflector is used to reduce a full mesh of BGP sessions among PE devices to a single BGP session between a PE and the route reflector.

- Autodiscovery with BGP is used to discover PE devices participating in a given VPN, PE devices participating in a given redundancy group, tunnel encapsulation types, multicast tunnel type, multicast members, etc.
- All-Active multihoming is used. This allows a given CE device to have multiple links to multiple PE devices, and traffic traversing to-and-from that CE fully utilizes all of these links (Ethernet segment).
- When a link between a CE device and a PE device fails, the PE devices for that EVPN instance (EVI) are notified of the failure with the withdrawal of a single EVPN route. This allows those PE devies to remove the withdrawing PE device as a next hop for every MAC address associated with the failed link (mass withdrawal).

BGP VPLS to EVPN Migration

Some service providers want to preserve their investments in VPLS. This leads to the need to connect the old VPLS networks to new networks that run EVPN. For this purpose, logical tunnel interfaces on the interconnection point of the VPLS and EVPN routing-instances was used. However, all the other PE devices either belonged to the VPLS network or the EVPN network and were unaware of the other technology.

Starting in Junos OS Release 18.1, EVPN can be introduced into an existing BGP VPLS network in a staged manner, with minimal impact to VPLS services. On a BGP VPLS PE device, some customers could be moved to EVPN, while other customers continue to use VPLS pseudowires. Other PE devices could be entirely VPLS and switching customers on other PEs to EVPN.

The seamless migration from BGP VPLS to EVPN solution supports the following functionality:

- Allow for staged migration toward EVPN on a site-by-site basis per VPN instance. For instance, new EVPN sites to be provisioned on EVPN PE devices.
- Allow for the coexistence of PE devices running both EVPN and VPLS for the same VPN instance and single-homed segments.

In the BGP VPLS to EVPN migration, the PE device where some customers have been migrated to EVPN while other customers are being served using VPLS, is called a super PE. As super PE devices discover other super PE devices within a routing instance, they use the EVPN forwarding to communicate with other super PE devices and VPLS pseudowires to PE devices running VPLS. The PE device with no EVPN awareness, and running only VPLS for all the customers, is called a VPLS PE.

The CE device connected to a super PE can reach CE devices connected to EVPN-only PE devices or VPLS-only PE device, but CE devices connected to EVPN-only PE devices cannot reach CE devices connected to VPLS-only PE devices.

Because the migration from BGP VPLS to EVPN is supported at a per routing instance basis, and if the routing instance is serving multiple customers on a PE device, all are migrated together. EVPN is responsible for setting up data forwarding between the PE devices upgraded to EVPN, while VPLS continues to setup data forwarding to PE devices that run VPLS.

NOTE:

The following features are not supported with the BGP VPLS to EVPN migration:

- Migration from FEC129 VPLS to EVPN.
- Migration of VPLS virtual switch to EVPN virtual switch.
- Migration of VPLS routing instance to EVPN virtual switch.
- Migration of VPLS routing instance or PBB-VPLS to PBB-EVPN.
- Seamless migration from EVPN back to VPLS.
- Enhancing EVPN to support the set of tools or statements and commands that VPLS supports.
- Active-active and active-standby multihoming. The migration to EVPN is supported only on single-homed deployments.
- Spanning all-active across EVPN and VPLS PE devices does not work, as all-active multihoming feature is not supported on VPLS.
- Connecting EVPN-only PE devices with VPLS-only PE devices through super PE devices.
- IPv6, logical systems, multi-chassis support, and SNMP, as they are currently not supported on EVPN.

EVPN Migration Configuration

To perform the BGP VPLS to EVPN migration, do the following:

1. On the backup Routing Engine, load Junos OS Release 18.1R1.
2. Perform ISSU to acquire mastership. Ensure that the VPLS ISSU does not have any impact on the VPLS forwarding.
3. Identify routing instances (customers) that need to be migrated to EVPN.
4. Enable family EVPN signaling in BGP by adding **family evpn** and **signaling** at the **[edit protocols bgp group-session]** hierarchy level.

   NOTE: Adding **family evpn** to the BGP protocol will cause the CE IFL to be deleted and recreated, so you should expect some traffic loss after this step.

```
protocols {
  bgp {
    group session {
```

```
        family evpn {
        signaling;
        }
    }
  }
}
```

5. Configure the ESI interface using a preference value that reflects the configured VPLS preference.

NOTE: Creating the ESI interface will cause the CE IFL to be deleted and recreated, so you should expect some traffic loss after this step.

```
[edit interfaces interface-name]
esi {
  00:01:02:03:04:05:06:00:00:01;
    single-active;
    df-election-type {
      preference {
        value match vpls preference;
        }
      }
  }
```

6. Enable EVPN in a single routing instance.

   - Change routing instance type from **vpls** to **evpn**, at the **[edit routing-instances *routing-intance-name*]** hierarchy level in your existing BGP VPLS configuratoin.

```
[edit routing-instances routing-instance-name]
instance-type evpn;
```

7. Include the **evpn** and **vpls** statements at the **[edit routing-instances *routing-intance-name* protocols]** hierarchy level in order to support EVPN and VPLS commands.

```
[edit routing-instances routing-instance-name]
protocols {
  evpn
  vpls
}
```

8. Once all nodes in the VPLS network have been migrated to EVPN, you can optionally decommission the VPLS protocol. This will allow EVPN to setup its single-homing and multi-homing state cleanly, but may result in traffic loss. To decommission the VPLS protocol, delete the **protocols vpls** statement under **[edit routing-instances *routing-instance-name*]** hierarchy.

```
[edit routing-instances routing-instance-name]
user@host# delete protocols vpls
```

After the configuration for the EVPN migration is committed, the routing protocol process and the Layer 2 address learning process start building the EVPN state to reflect interfaces, bridge domains, peers and routes. The locally learnt MAC addresses are synchronized by the Layer 2 address learning process in the instance.vpls.0 to the routing protocol process. When a local MAC ages out in the instance.vpls.0, the routing protocol process is informed by the Layer 2 address learning process.

When an EVPN peer is learnt, the routing protocol process sends a new message to the Layer 2 address learning process to remove the peer's label-switched interface or virtual tunnel logical interface from the VE mesh group and disables MAC-learning on it. The EVPN IM next-hop is then added to the VE mesh group. The EVPN behavior in the routing protocol process of learning MAC addresses over BGP and informing Layer 2 address learning process of the MPLS next hop is maintained.

The VPLS statements and commands continue to apply to the VPLS pseudowires between the PE devices and the MAC addresses learnt over them. The EVPN statements and commands apply to PE devices running EVPN.

Reverting to VPLS

If the EVPN migration runs into issues, you can revert back to VPLS until the issue is understood. The routing instance is reverted from a super PE to a VPLS PE in a non-catastrophic manner by enabling the following configuration:

```
[edit routing-instances routing-instance-name]
user@host# set instance-type vpls
user@host# delete protocols evpn
```

On reverting the EVPN migration to VPLS, the following happens:

1. The EVPN state information is deleted.
2. There is a trigger for withdrawal of EVPN control plane routes.
3. The routing protocol process sends a new message to the Layer 2 address learning process with the label-switched interface or the virtual tunnel logical interface for the routing instance and peer.
4. The label-switched or virtual tunnel interface adds the new message to the flood group and MAC learning is enabled.

5. The egress IM next hop is deleted by the routing protocols process, prompting the Layer 2 address learning process to remove it from the flood group.

6. Remote MAC addresses are learnt again over the label-switched interface or virtual tunnel logical interface.

BGP VPLS to EVPN Migration and Other Features

Table 4 on page 26 describes the functionality of some of the related features, such as multihoming and integrated routing and bridging (IRB) with the BGP VPLS to EVPN migration.

Table 4: EVPN Migration and Other Features Support

Feature	Supported Functionality in EVPN Migration
MAC move	MAC moves are supported between VPLS-only PE and super PE devices. When a MAC address moves from a VPLS-only PE to a super PE, it is learnt over BGP, and the routing protocol process informs the Layer 2 address learning process of the EVPN next hop to be updated in the foo.vpls.0 routing table. When a MAC address moves from a super PE to a VPLS-only PE, it is learnt in the Packet Forwarding Engine on the label-switched interface or virtual tunnel interface. The Layer 3 address learning process updates it to VPLS or the label-switched interface next hop. When the type 2 route is withdrawn by EVPN BGP, the MAC address is not deleted from the forwarding table, so there is no loss of data. The forwarding MAC table is shared by VPLS and EVPN. Some attributes, such as **mac-table-size** and **mac-table-aging-time** could be configured under both EVPN and VPLS. When there is a conflict, the values under EVPN take precedence.
IRB	No changes needed in IRB. On super PE, EVPN populates the /32 host routes learnt over MAC+IP type 2 routes from EVPN peers in a Layer 3 virtual routing and forwarding, while VPLS IRB forwarding using subnet routes work on sites still running VPLS.
Hierarchical VPLS	In a H-VPLS network with hub and spoke PE devices, when the hub PE is migrated to EVPN, local MAC addresses learnt over the access label-switched or virtual tunnel interface need to be advertised into BGP, so that the other EVPN-only PE devices or super PE devices can reach them. Take the following into consideration when migrating a H-VPLS network to EVPN: • Hubs typically have local-switching enabled as inter-spoke traffic is forwarded through the hub. If spoke(s) alone is migrated to EVPN and spokes have Layer 3 or MPLS reachability to each other, the label-switched or virtual tunnel interface to the hub and EVPN next hop (remote spoke) is present in the VE floodgroup and this results in two copies of broadcast, unknown unicast, and multicast (BUM) traffic received by the remote spoke. An option to avoid this is to migrate the hub(s) to EVPN too. • EVPN is not aware of hierarchy. All peers are considered core-facing. Once hubs and spokes are migrated to EVPN, split horizon prevents the BUM traffic from being forwarded to other core-facing PE devices.
ESI configuration	ESI is configured at the physical interface or port level.

Related Documentation

- EVPN Overview on page 549

CHAPTER 2

Configuring EVPN Multihoming

EVPN Multihoming Overview

Introduction to EVPN Multihoming

An Ethernet VPN (EVPN) comprises of customer edge (CE) devices that are connected to provider edge (PE) devices, which form the edge of the MPLS infrastructure. A CE device can be a host, a router, or a switch. The PE devices provide Layer 2 virtual bridge connectivity between the CE devices. There can be multiple EVPNs in the provider network. Learning between the PE routers occurs in the control plane using BGP, unlike traditional bridging, where learning occurs in the data plane.

NOTE: In releases earlier than Junos OS Release 15.1, EVPN functionality support on MX Series routers was limited to routers using MPC and MIC interfaces only. Starting with Junos OS Release 15.1, MX Series routers using DPCs can be leveraged to provide EVPN support on the CE device-facing interface.

DPC support for EVPN is provided with the following considerations:

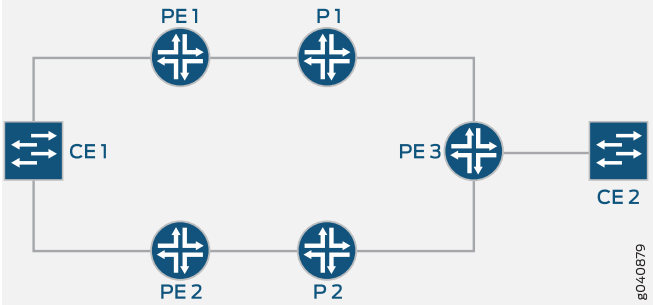
- DPCs provide support for EVPN in the active-standby mode of operation including support for the following:
  - EVPN instance (EVI)
  - Virtual switch
  - Integrated routing and bridging (IRB) interfaces
- DPCs intended for providing the EVPN active-standby support must be the CE device-facing line card. The PE device in the EVPN domain must be MPC interfaces or MIC interfaces.

The EVPN multihoming feature enables you to connect a customer site to two or more PE devices to provide redundant connectivity. A CE device can be multihomed to different PE devices or the same PE device. A redundant PE device can provide network service to the customer site as soon as a failure is detected. Thus, EVPN multihoming helps to maintain EVPN service and traffic forwarding to and from the multihomed site in the event of the following types of network failures:

- PE device to CE device link failure
- PE device failure
- MPLS-reachability failure between the local PE device and a remote PE device

Figure 3 on page 31 illustrates how a CE device can multihomed to two PE routers. Device CE 1 is multihomed to Routers PE 1 and PE 2. Device CE 2 has two potential paths to reach Device CE 1, and depending on the multihoming mode of redundancy, only one path or both the paths are active at any time. The multihoming mode of operation also determines which PE router or routers forward traffic to the CE device. The PE router forwarding traffic to the CE device (also called a designated forwarder) uses MPLS LSP or GRE tunnels to forward traffic. If a failure occurs over this path, a new designated forwarder is elected to forward the traffic to Device CE 1.

Figure 3: CE Device Multihomed to Two PE Routers



EVPN MPLS Multhoming Features Supported by QFX10000 Switches

Starting in Junos OS 17.4R1, QFX10000 switches support multihoming for EVPN MPLS. Only active-active multihoming is supported. The following subfeatures are supported:

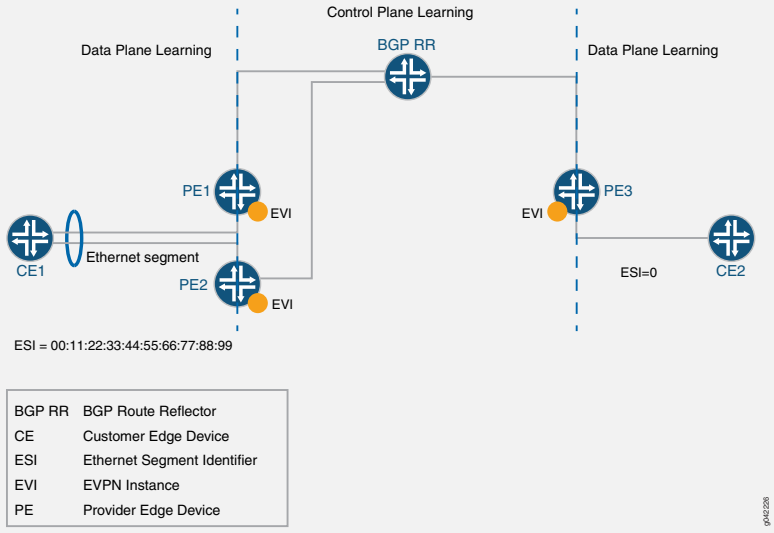
- ESI configuration (only type 0 manual configuration and IFD (physical interfaces) are supported)
- Aliasing and label route
- EVPN route type 4 (Ethernet segment route)
- Extended communities
- BUM traffic
- Designated Forwarder Election (DF) roles: DFand BDF

QFX10000 switches over an MPLS EVPN core only support the default-switch routing instance. An EVPN instance (EVI) is not supported.

Understanding EVPN Multihoming Concepts

Figure 4 on page 32 shows a simple EVPN network topology to define EVPN multihoming concepts. .

Figure 4: Simple EVPN Topology



- **Ethernet segment**—When a CE device is multihomed to two or more PE routers, the set of Ethernet links constitutes an Ethernet segment. An Ethernet segment appears as a link aggregation group (LAG) to the CE device.

  The links from Routers PE1 and PE2 to Device CE1 form an Ethernet segment.

  In active-standby multihoming, the links that constitute an Ethernet segment form a bridge domain. In active-active multihoming, an Ethernet segment appears as a LAG to the CE device.

- **ESI**—An Ethernet segment must have a unique nonzero identifier, called the Ethernet segment identifier (ESI). The ESI is encoded as a 10-octet integer. When manually configuring an ESI value, the most significant octet, known as the type byte, must be 00. When a single-homed CE device is attached to an Ethernet segment, the entire ESI value is zero.

  The Ethernet segment of the multihomed Device CE1 has an ESI value of 00:11:22:33:44:55:66:77:88:99 assigned. The single-homed Device CE2 has an ESI value of 0.

- **EVI**—An EVPN instance (EVI) is an EVPN routing and forwarding instance spanning all the PE routers participating in that VPN. An EVI is configured on the PE routers on a per-customer basis. Each EVI has a unique route distinguisher and one or more route targets.

  An EVI is configured on Routers PE1, PE2, and PE3.

- **Ethernet tag**—An Ethernet tag identifies a particular broadcast domain, such as a VLAN. An EVPN instance consists of one or more broadcast domains. Ethernet tags are assigned to the broadcast domains of a given EVPN instance by the provider of that EVPN. Each PE router in that EVPN instance performs a mapping between broadcast domain identifiers understood by each of its attached CE devices and the corresponding Ethernet tag.
- **Ethernet segment route**—The PE routers that are connected to a multihomed CE device use BGP Ethernet segment route messages to discover that each of the PE routers is connected to the same Ethernet segment. The PE routers advertise the Ethernet segment route, which consists of an ESI and ES-import extended community.

  Routers PE1 and PE2 advertise an ES route with an ES-import extended community (along with other extended communities like the route target). The PE routers also construct a filter that is based on an ES-import extended community, which results in only these PE routers importing the ES route and identifying that they are connected to the same Ethernet segment.
- **Extended community**— An extended community is similar in most ways to a regular community. EVPNs use extended communities because the 4-octet regular community value does not provide enough expansion and flexibility. An extended community is an 8-octet value divided into two main sections.
- **BUM traffic**—This type of traffic is sent to multiple destinations, including broadcast traffic, unknown unicast traffic that is broadcast in the Ethernet segment, and multicast traffic.
- **DF**—When a CE device is multihomed to two or more PE routers, either one or all of the multihomed PE routers are used to reach the customer site depending on the multihoming mode of operation. The PE router that assumes the primary role for forwarding BUM traffic to the CE device is called the designated forwarder (DF).
- **BDF**—Each router in the set of other PE routers advertising the autodiscovery route per Ethernet segment for the same ESI, and serving as the backup path in case the DF encounters a failure, is called a backup designated forwarder (BDF). A BDF is also called a non-DF router.
- **DF election**—On every Ethernet segment, the PE routers participate in a procedure called *designated forwarder* election to select the DF and the BDF PE routers.

EVPN Multihoming Mode of Operation

The different modes of operation for EVPN multihoming include:

- Single—When a PE router is connected to a single-homed customer site, this mode is in operation. The *single* mode is the default mode of operation, and does not require Ethernet segment values to be configured.
- Active-standby—When only a single PE router, among a group of PE routers attached to an Ethernet segment, is allowed to forward traffic to and from that Ethernet segment, the Ethernet segment is defined to be operating in the *active-standby* redundancy mode.

To configure the active-standby mode, include the ESI value and the **single-active** statement under the **[edit interfaces]** hierarchy level.

- Active-active—When all PE routers attached to an Ethernet segment are allowed to forward traffic to and from the Ethernet segment, the Ethernet segment is defined to be operating in the *active-active* redundancy mode.

  NOTE: In Junos OS Release 14.2 and earlier, the EX9200 Series switch supports only the active-standby mode of operation for EVPN multihoming.

  NOTE: Starting with Junos OS Release 14.1x53-D30 for QFX5100 switches and Junos OS Release 18.2R1 for EX4600 switches, these switches support the active-active mode of operation for EVPN multihoming. In this scenario, QFX5100 and EX4600 switches function as top-of-rack (ToR) switches in the data center for virtual networks. EVPN multihoming active-active functionality is used to provide access to the bare-metal servers connected to the top-of-rack switches.

  NOTE: Starting with Junos OS Release 14.1R4, 14.2, 15.1F6, and 16.1R1, Junos OS supports the active-active mode for EVPN multihoming on MX Series routers.

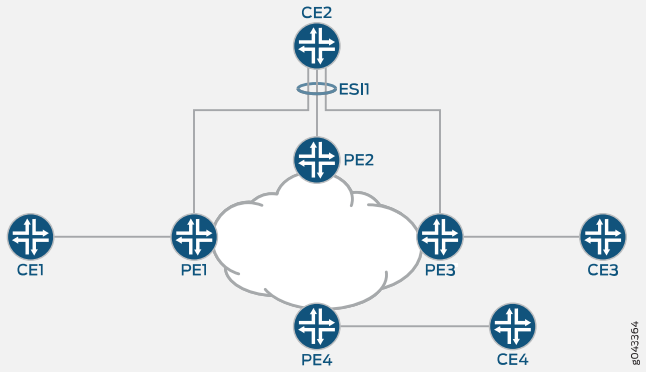
  Starting with Junos OS Releases 16.1R4 and 16.2R2, all EX9200 switches support the active-active mode for EVPN multihoming.

  Starting with Junos OS Releases 17.4R1 QFX10000 switches support the active-active mode for EVPN multihoming.

  To configure the active-active mode, include the ESI value and the **all-active** statement at the **[edit interfaces]** hierarchy level.

  Figure 5 on page 35 shows a reference topology for EVPN active-active multihoming. The ESI1 Ethernet segment for Device CE2 is multihomed to Routers PE1, PE2, and PE3. The Ethernet segment on the CE device can either be configured as a link aggregation group (LAG) or as an ECMP path. Devices CE1 and CE3 are the single-homed customer edge devices and have an ESI value of 0.

Figure 5: Active-Active EVPN Multihoming



EVPN Multihoming Implementation

The EVPN active-standby multihoming mode of operation provides redundancy for access link failures and PE node failure for the multihomed CE device, and is based on the EVPN *draft-ietf-l2vpn-evpn-03*.

The Junos OS implementation of the EVPN multihoming active-standby and active-active modes of operation includes the following:

New BGP NLRIs

To support EVPN multihoming, the following new BGP network layer reachability information (NLRI) routes have been introduced:

Autodiscovery Route per Ethernet Segment

Autodiscovery Route Features

The autodiscovery route NLRI features include:

- This is a Type 1 mandatory route, used for fast convergence and for advertising the split horizon label. It is also known as the mass withdraw route.
- Type 1 route distinguishers are used with the IP address (loopback) of the originating PE router as the route distinguisher value.
- This route carries the ESI in the NLRI (nonzero when it is a multihomed PE, zero otherwise).
- The split horizon label is per ESI only, and carries an explicit NULL (0).
- The bit in the active-standby flag field in the ESI label extended community is used for signaling the active-standby mode (bit set).
- The 3-byte label values in the NLRI and the Ethernet tag is zero.
- This route is advertised and imported by all multihomed and remote PE routers that share the same EVI on the advertising ESI.

Autodiscovery Route Advertisement

- Active-standby mode

  In active-standby mode, the designated forwarder (DF) advertises the autodiscovery route per Ethernet segment with an ESI MPLS label extended community that has the standby bit set to 1. The autodiscovery route is advertised per ESI, and the ESI label is set to 0 when active-standby mode is in operation.

  The autodiscovery route is imported by all the multihomed and remote PE routers that are part of the EVI. On receiving the autodiscovery route, the PE routers in the network topology learn that active-standby multihoming mode is in operation for the ESI advertised.

- Active-active mode

  In active-active mode, each of the multihomed PE device advertises a mandatory autodiscovery route per Ethernet segment as in the active-standby state. However, in the active-active state, the autodiscovery route per Ethernet segment is modified such that the active-standby bit carried in the MPLS extended community is cleared to indicate that the active-active mode is in operation. The autodiscovery route per Ethernet segment in the active-active mode also includes the split horizon label.

  In Figure 5 on page 35, for the ESI1 Ethernet segment, Routers PE1, PE2, and PE3 advertise the autodiscovery route. Router PE4 receives this autodiscovery route.

Autodiscovery Route Withdrawal

The autodiscovery route per Ethernet segment withdrawal may result in mass withdrawal. The mass withdrawal feature is used when there is a link failure on the ESI, or when the ESI configuration changes.

When the link between a multihomed CE device and a multihomed PE device fails, the PE device withdraws the autodiscovery route per Ethernet segment. In such a case, the mass withdrawal feature is handled in the following ways by the other PE devices:

- Remote PE device

  When a remote PE device receives the BGP update for mass withdrawal, the following is performed at the remote PE device:

  1. The current next hop to reach the remote ESI or CE device is deleted.
  2. A new next hop through the remaining multihomed PE devices is created to reach the remote ESI or CE device.
  3. All the MAC routes behind the CE device are updated with the newly created next hop.

  Starting with Junos OS Release 17.4R1, Junos OS supports Dynamic List Next Hops in an EVPN network. Now when the link between the CE device and a multihome PE device fails, the next hop to the ESI or CE is updated, thus reducing the need for a mass withdrawal. For more information on enabling Dynamic List Next Hop, see "Configuring Dynamic List Next Hop" on page 211.

- Other multihomed PE device

  As a result of the mass withdrawal, load balancing on the multihomed CE device happens because of the following:

  - When the other multihomed PE devices receive the same set of MAC addresses on the link to the concerned ESI.

    In this case, the local routes are preferred. If the remote routes learned from the DF PE device gets withdrawn, it does not affect routes pointing to the local ESI.

  - When the other multihomed PE devices have not received the same set of MAC addresses on the link to the concerned ESI.

    In this case, the PE devices install the MAC routes pointing to the concerned ESI, although the MACs are remotely learned from the DF PE device. When the DF PE device withdraws these routes, the withdrawn routes are flushed. Packets that are destined to the flushed MAC addresses are flooded on all the local segments.

Ethernet Segment Route

- Ethernet Segment Route Features on page 38
- Ethernet Segment Route Advertisement on page 38

Ethernet Segment Route Features

The Ethernet segment route NRLI features include:

- This is a Type 4 route. The purpose of this route is to enable the PE routers connected to the same Ethernet segment to automatically discover each other with minimal configuration on exchanging this route.
- This route is associated with an ES-import extended community with an ESI value condensed to 6 bytes, similar to a route target.
- This route is advertised and imported only by PE routers that are multihomed on the advertising Ethernet segment.

Ethernet Segment Route Advertisement

The Ethernet segment route is exchanged among all the PE routers within a data center with the ES-import extended community. The ES-import extended community is constructed based on the ESI PE routers that are multihomed, and the Ethernet segment route carries the ESI value related to the Ethernet segment on which the PE routers are multihomed.

The Ethernet segment routes are filtered based on the ES-import extended community, such that only the PE routers that are multihomed on the same Ethernet segment import this route. Each PE router that is connected to a particular Ethernet segment constructs an import filtering rule to import a route that carries the ES-import extended community.

Autodiscovery Route per EVPN Instance

In active-active mode, each of the multihomed PE devices advertise an autodiscovery route per EVPN instance (EVI) with a valid MPLS label. This route is advertised per ESI and is imported by the remote PE devices. The MPLS label included in the autodiscovery route per EVI is used later for aliasing.

New Extended Communities

An extended community is similar in most ways to a regular community. Some networking implementations, such as virtual private networks (VPNs), use extended communities because the 4-octet regular community value does not provide enough expansion and flexibility. An extended community is an 8-octet value divided into two main sections.

To support active-standby multihoming, the following extended communities have been introduced:

- ESI-Import on page 38
- Split Horizon on page 39

ESI-Import

This extended community is attached to the ES route, and is populated from the ESI-import value extracted from the configured ESI value under the interface. To solve the problem of a conflict with another regular route target, the type is set to 0x06, which has been allocated by IANA.

The ESI-import extended community route target populates the list of import route targets configured for the special instance from where the ES route using this community is advertised.

Therefore, incoming ESI routes with the same ESI-import value in the extended community are imported by the PE routers, if the PE router is configured with an Ethernet segment that has the same ESI value. Once the PE router receives a set of these ESI routes that have the same ESI-import extended community value, the DF and BDF election can be done locally.

NOTE: **When the ESI-import extended community is not created implicitly, a policy must be configured to attach all the route targets to the autodiscovery route per Ethernet segment.**

Split Horizon

With reference to Figure 5 on page 35 for example, when a CE device that is multihomed to two or more PE devices on an Ethernet segment (ESI1) and operating in the active-active redundancy mode sends a BUM packet to one of the non-DF PE devices (say PE1), then Device PE1 forwards that packet to all or a subset of the other PE devices in that EVPN instance, including the DF PE device for that Ethernet segment. In this case the DF PE device that the CE device is multihomed to drops the packet without forwarding it back to the CE device. This filtering is referred to as split horizon.

- Split horizon signaling

  The split horizon extended community is attached to the autodiscovery route per Ethernet segment. The value of the extended community is the split horizon or the Poisson label itself, which is 3 bytes, and is advertised as an opaque attribute.

- Split horizon advertisement
  - In active-standby mode, the standby bit in the split horizon extended community is set to 1, and the ESI split horizon label is set to 0.
  - In the active-active mode, the split horizon extended community is modified to clear the standby bit to 0 and includes a valid ESI label used for split horizon purposes.

- Split horizon MPLS routes

  The DF PE device advertises an autodiscovery route per Ethernet segment with a split horizon label A, and an inclusive multicast route with label B for BUM traffic forwarding. On the DF, the BUM packet from the core can come with following labels:

  - When the non-DF PE devices receive a BUM packet on their single-homed ESIs, the BUM packet is sent to the DF PE device with multicast label B.
  - When the non-DF PE devices receive a BUM packet on ESI1, the BUM packet is sent to the DF PE device with two MPLS labels — the multicast label B as the outer label, and the split horizon label A as the inner label.

  In the EVPN multihoming scenario, the multicast label B has the S-bit set to 1 when it is the only label in the label stack. In this case, the BUM packet needs to be flooded on

all the local ESIs on the DF PE device. But the label B has the S-bit set to 0 when split horizon label A is the innermost label in the label stack. In this case, the BUM packets need to be flooded on all local ESIs on the DF PE device, except the ESI that maps to the split horizon label A.

Assuming that packets originated from a multihomed CE device to a non-DF PE device on multihomed segment ESI1, when the non-DF PE device sends this packet to the DF PE device, the ESI label that the DF advertised to the non-DF PE device in its autodiscovery route per Ethernet segment is pushed first. The non-DF PE device also pushes the inclusive multicast label that the DF PE device advertised in its inclusive multicast route and further pushes the LSP label. The MPLS header thus contains two labels within a 32-bit field.

The base EVPN functionality uses a table-next hop to stitch the MPLS table with its corresponding EVPN EVI table. In the EVPN EVI table, the mac-lookup is performed to switch the packet.

The following routes are programmed in the mpls.0 table for EVPN multicast:

- The (multicast-label, S=1) route points to the EVPN-EVI table-next hop.
- The (multicast-label, S=0) route points to the MPLS table-next hop. This route loops the packet back to the MPLS table after popping the multicast-label.
- The (split horizon-label) route points to the EVPN-EVI table-next hop. This is the same table-next hop that is used by the multicast-label, S=1 route.

New EVPN Route Types

EVPN multihoming mode supports the following EVPN route types:

- Autodiscovery route per Ethernet segment
- Autodiscovery route per EVPN instance (EVI)
- Ethernet segment route

These route types conform to the following naming convention:

<route-type>:<RD>::<esi>::<route-specific>/304

For example:

Autodiscovery route per Ethernet segment—**1:10.255.0.2:0::112233445566778899::0/304**

Autodiscovery route per EVI—**1:100.100.100.1:1::2222222222222222222::0/304**

Ethernet segment route—**4:10.255.0.1:0::112233445566778899:10.255.0.1/304**

where:

- **route-type**—Type of EVPN route.
  - 1—Autodiscovery route per Ethernet segment.
  - 1—Autodiscovery route per EVI.

  - 4—Ethernet segment route.
  - 5—Route with VXLAN/MPLS encapsulation

- **RD**—Route distinguisher value.

  The route distinguisher value is set to the IP address of the PE router followed by 0.

- **esi**—Ethernet segment identifier. Displayed as 10 bytes of hexadecimal bytes, and leading 00 bytes are not displayed.
- **route-specific**—Differs per route type.
  - Autodiscovery route per Ethernet segment and autodiscovery route per EVI—This value is an MPLS label.

    NOTE: **The MPLS label is displayed in the extensive output, although it is not included in the prefix.**

  - Ethernet segment route—This value is the originating IP address.
- **304**—Maximum number of bits in an EVPN route. This is not very useful information and could be removed from the display. However, it might be useful in quickly identifying an EVPN route, either visually or with match operators.

Update to the MAC Forwarding Table

In active-standby EVPN multihoming, the MAC addresses are treated as routable addresses, and the MP-IBGP protocol is used to carry the customer MAC addresses. MAC learning at the PE routers does not occur in the data plane but in the control plane. This leads to more control applied in terms of the learning mechanism.

A PE router performs MAC learning in the data plane for packets coming from a customer network for a particular EVI. For CE MAC addresses that are behind other PE routers, the MAC addresses are advertised in BGP NLRI using a new MAC advertisement route type.

The MAC learning is of two types:

- Local MAC learning—PE routers must support the local MAC learning process through standard protocols.
- Remote MAC learning—Once the local learning process is completed, the PE routers can advertise the locally learned MAC address to remote PE router nodes through MP-IBGP. This process of receiving the remote MAC addresses of attached customers through MP-IBGP is known as the remote MAC learning process.

The MAC advertisement route type is used to advertise locally learned MAC addresses in BGP to remote PE routers. If an individual MAC address is advertised, the IP address field corresponds to that MAC address. If the PE router sees an ARP request for an IP address from a CE device, and if the PE router has the MAC address binding for that IP address, the PE router performs ARP proxy and responds to the ARP request.

NOTE: The ARP proxy is performed only for the gateway and not for the host.

The MPLS label field depends on the type of allocation. The PE router can advertise a single MPLS label for all MAC addresses per EVI, which requires the least number of MPLS labels and saves the PE router memory. However, when forwarding to the customer network, the PE router must perform a MAC lookup which can cause a delay and increase the number of CPU cycles.

Traffic Flow

In EVPN multihoming, traffic flow is performed in the forwarding-plane. Flood routes are created for flooding the packets, and are used in the following scenarios:

- When a packet is received on a local ESI
- When a packet is received from the core

The traffic flows in EVPN multihoming can be based on the two traffic types:

- Unicast traffic

  Unicast traffic is a point-to-point communication with one sender and one receiver. In a multihomed EVPN, unicast traffic is forwarded as follows:

  - In active-standby mode
    - CE to core—Traffic is learned and forwarded by the DF PE router.
    - Core to CE—The remote PE router learns the MAC addresses from the DF, and forwards all unicast traffic to the DF PE router.
  - In active-active mode
    - CE to core—Traffic is load-balanced to all the connected multihomed PE devices.
    - Core to CE—Traffic from the remote PE devices is load-balanced to all the multihomed PE devices connected to the remote CE device.

- BUM traffic

  Traffic that is sent to multiple destinations, including broadcast traffic, unknown unicast traffic that is broadcast in the Ethernet segment, and multicast traffic is known as BUM traffic. In a multihomed EVPN, BUM traffic is forwarded as follows:

  - In active-standby mode
    - CE to core—The CE device floods any BUM traffic to all the links in the Ethernet segment. The DF PE router with the active path forwards the BUM packets to the core. The BDF PE router in the standby mode drops all the traffic from the CE device, because the EVPN multihomed status of the interface is in blocking state. However, if the CE device is connected to the PE devices using separate links or LAGs, the BUM traffic reaches both the DF and BDF PE devices.

- Core to CE—The remote PE routers flood all BUM traffic to both the DF and BDF PE routers. Only the DF forwards the BUM traffic to the CE device. The BDF PE router drops all the traffic, because the EVPN multihomed status of the interface is in blocking state.

- In active-active mode

  Based on the requirements, flooding and switching among local ESIs can be enabled or disabled in the active-active mode. This is referred to as the no-local-switching behavior.

  The core of EVPN service provides a full-mesh connectivity among the multihomed PE devices. Because of this, EVPN uses split horizon in the core, so a packet received from the core is never switched or flooded back to the core. Instead, ingress replication is used to replicate the packets to the remote PE devices.

  To flood packets to remote PE devices, the multicast and the split horizon next hops are used. The multicast next hop tunnels the packet with the inclusive multicast label, and the split horizon next hop tunnels the packet with a multicast-label and a split horizon label. One such next hop is required per multihomed ESI per remote PE device.

  The following flood routes are used in the active-active mode:

  - All-CE flood route

    This flood route is used by the local ESIs for the following:

    - Flooding the packet on the local ESIs (when local-switching is allowed).
    - Flooding the packet to the remote PE devices. The remote PE devices flood the packet on their local ESIs.

    Because BUM traffic is forwarded only by the Designated Forwarder (DF), and not by the non-DF multihomed PE devices, the non-DFs use the split horizon next hop to flood this packet to other PE devices. However, the multihomed local ESIs for which the PE device is a non-DF does not participate in the flooding.

    The all-CE flood route is not used by the non-DF ESIs, and the next hop for these flood routes is created accordingly. In such cases, the non-DF ESI flood route is used.

  - All-VE flood route

    This flood route is used when the packet is received from the core. It is used for flooding the packet received from the core to the local ESIs. Because the packet received from the core can come with multicast-label only or with both multicast-label and split horizon label, appropriate forwarding rules must be followed to drop the packet on the multihomed ESI that maps to the split horizon label.

  - Non-DF flood route

    This flood route is used for the following:

    - Flooding the packet on the local ESIs.

- Flooding the packet to the remote PE devices using ingress replication with SH-label for the DF for the ESI.

Aliasing

Starting in Junos OS Release 15.1, Junos OS supports aliasing in an EVPN. Aliasing is the ability of a remote PE device to load balance Layer 2 unicast traffic on all the other PE devices that have same Ethernet segment towards a CE device.

Aliasing in the Active-Active Mode

In Figure 5 on page 35, aliasing in the active-active mode works as follows:

1. ESI1 is configured on Routers PE1, PE2, and PE3. Routers PE1, PE2, and PE3 advertise the autodiscovery route per Ethernet segment for ESI1.
2. Device CE1 sends Layer 2 traffic with source MAC address (MAC1) to Router PE1.
3. Router PE1 learns the MAC1 address on (ESI1, vlan X) and advertises it to all PE routers using BGP.
4. Router PE4 receives the MAC1 route through BGP.
5. Because Router PE4 also received the autodiscovery route per EVI from Routers PE2 and PE3, it knows that MAC1 must be reachable through Routers PE2 and PE3. Router PE4 builds its forwarding state to load-balance the Layer 2 traffic for MAC1 among Routers PE1, PE2, and PE3.

Aliasing and Autodiscovery Routes

Autodiscovery routes from Routers PE2 and PE3 can come in any order. As a result, these routes are installed by the Layer 2 process as follows:

1. After receiving MAC1 from Router PE1, and if any autodiscovery routes have not been received by Router PE4, MAC1 is programmed by PE4 with a next hop pointing toward Router PE1. When PE4 receives the autodiscovery route from Router PE2 for the same ESI, the next hop is installed so the traffic for MAC1 is load-balanced to Routers PE1 and PE2. When PE4 receives the autodiscovery route from Router PE3 for the same ESI, the next hop is updated to load-balance the traffic for MAC1 among Routers PE1, PE2, and PE3.

2. If Router PE4 has already received the autodiscovery routes from more than one PE device (PE1, PE2, and PE3), PE4 installs the MAC routes with the multi-destination next hop.

Aliasing and Label Route

Any PE device that advertises the autodiscovery route per EVI with a valid MPLS label programs the advertised label in the mpls.0 routing table. For instance, if Router PE2 advertised the autodiscovery route per EVI with label A, the mpls.0 entry is as follows:

Label A route points to the EVPN-EVI table-next hop.

When the remote Router PE4 sends a unicast data packet toward Router PE2 with this label A, lookup is done in Router PE2's forwarding table, and as a result of this lookup, the packet is forwarded on ESI1.

Aliasing and Unicast Packet Forwarding

When the unicast packets for MAC1 come from the remote Router PE4 to Router PE2, there could be two cases:

- Router PE2 also received the same set of MACs on its link to ESI1—In this case, local routes are preferred and as a result of the MAC lookup, packets are forwarded to ESI1.
- Router PE2 has not received the same set of MACs on its link to ESI1—In this case, Router PE2 still installs MAC routes pointing to ESI1, although MACs are remotely learned from Router PE1. As a result, the packets are forwarded to ESI1.

EVPN Active-Active Multihoming and Multichassis Link Aggregation

When a CE device is configured with a LAG toward the PE devices, the following two options are available to run LACP on the PE devices:

- Configure the same LACP system ID on all the PE devices.
- Configure multichassis link aggregation on the PE devices.

When multichassis link aggregation is configured with EVPN, a reduced set of procedures for active-active multichassis link aggregation are required. These procedures provide link and node level redundancy. The multichassis link aggregation is completely transparent to the CE device, and is realized as pure LAG. Multichassis link aggregation operates at the port level as well. This essentially means that if multichassis link aggregation is configured as active-active, all VLANs on the multichassis link aggregation ports work in the active-active multihoming mode.

When multichassis link aggregation is configured along with EVPN, the following is considered:

- Both multichassis link aggregation and EVPN ESI must be enabled to work in the active-active mode only.
- The following functions are not required for multichassis link aggregation with EVPN:
  - Mac synchronization—This is performed in the BGP control plane of EVPN.

- ICL linking—This is handled by the aliasing feature of EVPN.
- ARP synchronization—This is handled by the BGP control plane with IRB functionality.

EVPN Active-Active Multihoming and IRB

When IRB is configured, the EVPN routes contain both MAC and IP information. The active-active multihoming requires ARP synchronization among the multihomed PE devices because the ARP responses can get hashed to a particular PE device.

Sample Configuration

The following is a sample configuration for EVPN active-standby multihoming on the following types of interfaces:

- Ethernet interface configuration

```
ge-0/1/2 {
   encapsulation ethernet-bridge;
   esi XX:XX:XX:XX:XX:XX:XX:XX:XX:XX;
   unit 0 {
      family bridge;
   }
}
```

- Single VLAN interface configuration

```
ge-0/1/3 {
   encapsulation extended-vlan-bridge;
   esi XX:XX:XX:XX:XX:XX:XX:XX:XX:XX;
   vlan-tagging
   unit 0 {
      family bridge;
      vlan-id 1;
   }
}
```

NOTE:

- An ESI value of 0 and all FFs are reserved and are not used for configuring a multihomed Ehernet segment.
- Two interfaces in the same EVI cannot be configured with the same ESI value.

The following is a sample routing instance configuration for EVPN active-standby multihoming:

- Routing instance configuration

```
routing-instances {
   evpn-0 {
```

```
    instance-type evpn;
    route-distinguisher value;
    vrf-target value;
    vlan-id vlan-ID;
    interface ge-0/1/2.0;
    interface ge-1/1/1.0;
    interface ge-2/2/2.0;
    protocols {
      evpn {
        designated-forwarder-election hold-time time;
      }
    }
  }
}
```

NOTE: With the active-standby mode configuration, the autodiscovery route per Ethernet segment is advertised with the active-standby bit set to 1 for each Ethernet segment.

Designated Forwarder Election

The following sections discuss DF election:

DF Election Roles

The designated forwarder (DF) election process involves selecting the designated forwarder (DF) PE router and the backup designated forwarder (BDF) or a non-DF (non-designated forwarder PE router roles.

- **DF**—The MAC address from the customer site is reachable only through the PE router announcing the associated MAC advertisement route. This PE router is the primary PE router that is selected to forward BUM traffic to the multihomed CE device, and is called the designated forwarder (DF) PE router.
- **BDF**—Each PE router in the set of other PE routers advertising the autodiscovery route per Ethernet segment for the same ESI, and serving as the backup path in case the DF encounters a failure, is called a backup designated forwarder (BDF) or a non-DF (non-designated forwarder) PE router.

  As a result of the DF election process, if a local PE router is elected as the BDF, the multihomed interface connecting to the customer site is put into a blocking state for the active-standby mode. The interface remains in the blocking state until the PE router is elected as the DF for the Ethernet segment that the interface belongs to.

DF Election as Per RFC 7432

DF Election Procedure

The default procedure for DF election at the granularity of the ESI and EVI is referred to as service carving. With *service carving,* it is possible to elect multiple DFs per Ethernet segment (one per EVI) in order to perform load-balancing of multidestination traffic destined for a given Ethernet segment. The load-balancing procedures carve up the EVI space among the PE nodes evenly, in such a way that every PE is the DF for a disjoint set of EVIs.

The procedure for service carving is as follows:

1. When a PE router discovers the ESI of the attached Ethernet segment, it advertises an autodiscovery route per Ethernet segment with the associated ES-import extended community attribute.

2. The PE router then starts a timer (default value of 3 seconds) to allow the reception of the autodiscovery routes from other PE nodes connected to the same Ethernet segment. This timer value must be the same across all the PE routers connected to the same Ethernet segment.

   The default wait timer can be overwritten using the **designated-forwarder-election hold-time** configuration statement.

3. When the timer expires, each PE router builds an ordered list of the IP addresses of all the PE nodes connected to the Ethernet segment (including itself), in increasing numeric order. Every PE router is then given an ordinal indicating its position in the ordered list, starting with 0 as the ordinal for the PE with the numerically lowest IP address. The ordinals are used to determine which PE node is the DF for a given EVI on the Ethernet segment.

4. The PE router that is elected as the DF for a given EVI unblocks traffic for the Ethernet tags associated with that EVI. The DF PE unblocks multidestination traffic in the egress direction toward the Ethernet segment. All the non-DF PE routers continue to drop multidestination traffic (for the associated EVIs) in the egress direction toward the Ethernet segment.

In Figure 5 on page 35, the election of the DF for active-active multihoming is performed among Routers PE1, PE2, and PE3. As a result of this DF election, each one of these routers can become the DF for a particular VLAN from a range of VLANs configured on ESI1. The DF is responsible for forwarding BUM traffic on that ESI and VLAN for which it is elected as the DF. The non-DF PE routers block the BUM traffic on that particular Ethernet segment.

DF Election Trigger

In general, a DF election process is triggered in the following conditions:

- When an interface is newly configured with a nonzero ESI, or when the PE router transitions from an isolated-from-the-core (no BGP session) state to a connected-to-the-core (has established BGP session) state, a wait timer is imposed. By default, the interface is put into a blocking state until the PE router is elected as the DF.
- After completing a DF election process, a PE router receives a new Ethernet segment route or detects the withdrawal of an existing Ethernet segment route, without an imposed wait timer.
- When an interface of a non-DF PE router recovers from a link failure, the PE router has no knowledge of the wait time imposed by other PE routers. As a result, no wait timer is imposed for the recovered PE router to avoid traffic loss.

Preference-Based DF Election

The DF election based on RFC 7432 does not meet some of the operational requirements needed by some service providers. As a solution to this, starting with Junos OS Release 17.3, the DF election in a multihoming EVPN network can be controlled by using an administrative preference value for an ESI.

In the default DF election procedure (as specified in RFC 7432), the DF is elected randomly from one of the multihoming devices with modulo operation. With the preference-based DF election, the DF is elected manually using interface configuration options, such as the preference value, the Don't Preempt (DP) bit, and router ID or loopback address.

- Preference-Based DF Election Procedure on page 49
- DF Election Algorithm Mismatch on page 50
- DF Election Algorithm Migration on page 50
- Changing Preference for Maintenance on page 50

Preference-Based DF Election Procedure

The preference-based DF election is supported on EVPN and PBB-EVPN, and allows for manually electing a DF. This is useful when there is a need to choose the DF based on interface attributes, such as the bandwidth associated with an interface.

The preference-based DF election is executed as follows:

1. The DF election type and preference value are configured under an ESI. By default, the preference-based DF election type is based on the modulo (MOD) operation.

2. The configured preference value and DP bit are advertised to the multihoming PE devices using the DF election extended community in the type 4 routes.

3. After receiving the type 4 route, the PE device builds the list of candidate DF devices, in the order of the preference value, DP bit, and IP address.

4. When the DF timer expires, the PE device selects the DF based on the highest preference value.

   By default, the DF is elected based on highest preference per EVI. However, the preference-based DF election allows for electing the DF based on the lowest preference value when the **designated-forwarder-preference-least** statement is included at the **[edit routing-instances routing-instance-name protocols evpn]** hierarchy level.

   NOTE: The designated-forwarder-preference-least configuration should be the same on both the multihoming EVIs; otherwise there can be two DFs causing traffic loss or loop.

5. When the same preference value is configured, then the PE device selects the DF based on the DP bit. When the DP bit is also the same, the DF is elected based on the lowest IP address.

DF Election Algorithm Mismatch

When there is a mismatch between a locally configured DF election algorithm and a remote PE device's DF election algorithm, then all the PE devices should fall back to the default DF election as specified in RFC 7432.

DF Election Algorithm Migration

During the migration of the old DF election to the new DF election, it is expected to change the configuration during the maintenance window by bringing down the ESI, and changing the DF election algorithm.

To do the migration, do the following:

1. After a software upgrade, on the non-DF device bring down all the interfaces that have the same ESI.
2. Configure the new DF election algorithm on the DF PE.
3. Configure the DF election algorithm on other multihoming PE devices.
4. Bring up all the interfaces on the non-DF PE devices.

Changing Preference for Maintenance

After migrating the DF election algorithm, and all the multihoming PE device are running the preference-based DF election algorithm, maintenance tasks required on the existing DF can be executed by simply changing the configured preference value. This changes the DF for a given ESI.

To change the DF for a given ESI:

1. Change the preference value to a higher value on the current non-DF device.

2. Change the preference value to a lower value on the current DF device.

NOTE: Changing the preference value for an ESI can lead to some traffic loss during the short duration required to integrate the delay in the updated BGP route propagation with the new preference value.

DF Election for Virtual Switch

The virtual switch allows multiple bridge domains in a single EVPN instance (EVI). The virtual switch also supports trunk and access ports. Junos OS allows flexible Ethernet services on the port; therefore different VLANs on a single port can be part of different EVIs.

The DF election for virtual switch depends on the following:

- Port mode—Sub-interface, trunk interface, and access port
- EVI mode—Virtual switch with EVPN and EVPN-EVI

In the virtual switch, multiple Ethernet tags can be associated with a single EVI, wherein the numerically lowest Ethernet tag value in the EVI is used for the DF election.

Handling Failover

A failover can occur when:

- The DF PE router loses its DF role.
- There is a link or port failure on the DF PE router.

On losing the DF role, the customer-facing interface on the DF PE router is put in the blocking state.

In the case of link or port failure, a DF election process is triggered, resulting in the BDF PE router to be selected as the DF. At that time, unicast traffic and BUM flow of traffic are affected as follows:

- Unicast Traffic on page 51
- BUM Traffic on page 52

Unicast Traffic

- CE to Core—The CE device continues to flood traffic on all the links. The previous BDF PE router changes the EVPN multihomed status of the interface from the blocking state to the forwarding state, and traffic is learned and forwarded through this PE router.
- Core to CE—The failed DF PE router withdraws the autodiscovery route per Ethernet segment and the locally-learned MAC routes, causing the remote PE routers to redirect traffic to the BDF.

NOTE: The transition of the BDF PE router to the DF role can take some time, causing the EVPN multihomed status of the interface to continue to be in the blocking state, resulting in traffic loss.

BUM Traffic

- CE to Core—All the traffic is routed toward the BDF.
- Core to CE—The remote PE routers flood the BUM traffic in the core.

ESIs on Physical, Aggregated Ethernet, and Logical Interfaces

In releases before Junos OS Release 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI only on a physical or aggregated Ethernet interface, for example, **set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99**. If you specify an ESI on a physical or aggregated Ethernet interface, keep in mind that an ESI is a factor in the designated forwarder (DF) election process. For example, assume that you configure EVPN multihoming active-standby on aggregated Ethernet interface ae0, and given the ESI configured on ae0 and other determining factors, the DF election results in ae0 being in the down state. Further, all logical interfaces configured on aggregated Ethernet interface ae0, for example, **set interfaces ae0 unit 1** and **set interfaces ae0 unit 2** are also in the down state, which renders logical interfaces ae0.1 and ae0.2 unable to provide services to their respective customer sites (VLANs).

To better utilize logical interfaces in EVPN multihoming active-standby or active-active mode, starting with Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. As a result, even if a logical interface is a non-DF, other logical interfaces on the same physical or aggregated Ethernet interface are still able to provide services to their respective customer sites (VLANs).

For more information, see "Example: Configuring an ESI on a Logical Interface With EVPN Multihoming" on page 214.

Convergence in an EVPN Network

When there are changes in the network topology in a large-scale EVPN system, the convergence time might be significant. You can prioritize NLRI updates that are critical to route selection in routing policies to improve convergence. Table 5 on page 52 lists the NLRI route types and the priority that must be configured in the routing policy.

Table 5: Priority for NLRI Route Type

NLRI Route Type	Description	Priority
NLRI Route Type 1	Ethernet auto-discovery route—Type 1 supports fast convergence and aliasing and is used to signal MAC mass withdrawal.	High
NLRI Route Type 2	MAC/IP advertisement route—Type 2 is used to advertise MAC addresses and IP addresses in EVPN networks.	Low

Table 5: Priority for NLRI Route Type (continued)

NLRI Route Type	Description	Priority
NLRI Route Type 3	Inclusive multicast Ethernet tag—Type 3 is used to set up a path for BUM traffic.	Low
NLRI Route Type 4	Ethernet segment route—Type 4 is used in the selection of a designated forwarder.	High

To prioritize the NLRI route type, set the **bgp-output-queue-priority priority** for **nlri-route-type** at the [**edit policy-options policy-statement**] hierarchy level on all provider edge routers and route reflectors in the EVPN network. In this example, a high priority was configured for NRLI route type 1 and NRLI route type 4.

user@PE1#`show policy-options`

```
policy-statement evpn-rt-priority-policy {
    term 1 {
        from {
            family evpn;
            nlri-route-type 1;
        }
        then {
            bgp-output-queue-priority priority 16;
        }
    }
    term 2 {
        from {
            family evpn;
            nlri-route-type 2;
        }
        then {
            bgp-output-queue-priority priority 1;
        }
    }
    term 3 {
        from {
            family evpn;
            nlri-route-type 3;
        }
        then {
            bgp-output-queue-priority priority 2;
        }
    }
    term 4 {
        from {
            family evpn;
            nlri-route-type 4;
        }
        then {
            bgp-output-queue-priority priority 16;
        }
    }
}
```

NOTE: There are 17 prioritized output queues: an expedited queue that has the highest priority, and 16 numbered queues for which1 is the lowest priority and 16 is the highest.

For more information about how to configure routing policies, see *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*.

Release History Table

Release	Description
17.4R1	Starting with Junos OS Release 17.4R1, Junos OS supports Dynamic List Next Hops in an EVPN network.
16.1R4	Starting with Junos OS Releases 16.1R4 and 16.2R2, all EX9200 switches support the active-active mode for EVPN multihoming.
16.1R4	Starting with Junos OS Releases 17.4R1 QFX10000 switches support the active-active mode for EVPN multihoming.
15.1F6	To better utilize logical interfaces in EVPN multihoming active-standby or active-active mode, starting with Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. As a result, even if a logical interface is a non-DF, other logical interfaces on the same physical or aggregated Ethernet interface are still able to provide services to their respective customer sites (VLANs).
15.1	Starting in Junos OS Release 15.1, Junos OS supports aliasing in an EVPN.
14.1x53-D30	Starting with Junos OS Release 14.1x53-D30 for QFX5100 switches and Junos OS Release 18.2R1 for EX4600 switches, these switches support the active-active mode of operation for EVPN multihoming.
14.1R4	Starting with Junos OS Release 14.1R4, 14.2, 15.1F6, and 16.1R1, Junos OS supports the active-active mode for EVPN multihoming on MX Series routers.

Related Documentation

- Configuring Dynamic List Next Hop on page 211
- Example: Configuring EVPN Active-Standby Multihoming on page 79
- Example: Configuring EVPN Active-Active Multihoming on page 123

Configuring EVPN Active-Standby Multihoming to a Single PE Device

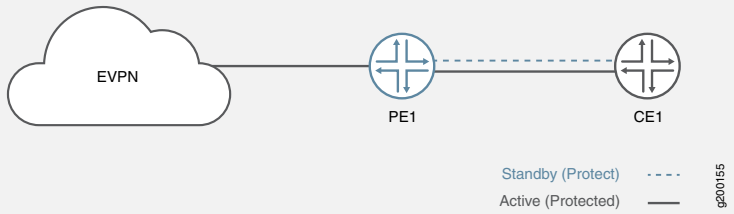
You can configure EVPN active-standby multihoming on a single PE device by configuring a standby (protect) interface to provide redundancy to an active (protected) interface. Figure 6 on page 55 illustrates an EVPN topology with active-standby multihoming to a single PE device. A protect interface provides the benefit of a backup for the protected primary interface in case of failure. The PE device uses the primary interface for network traffic while the primary interface is functioning. If the primary interface fails, the protect interface becomes active and traffic switches to the protect interface. When the primary interface returns, the primary interface becomes the active interface once again.

NOTE: If connectivity fault management (CFM) is enabled, the protect interface will trigger CFM to send Type, Length, and Value (TLV) or Remote Defect Indication (RDI) interface status to the customer edge device

Junos OS does not support the protect interface in the following cases:

- EVPN-VXLAN
- PBB-EVPN
- On an interface that has been configured as an ESI in EVPN multihoming

Figure 6: EVPN Active-Standby Multihoming on a Single PE Device



To configure the protect interface in a routing instance, configure both the protect interface and the primary interface in the routing instance for EVPN and include the **protect-interface** statement in the primary interface hierarchy.

```
routing-instances {
   routing-instance-name{
      instance-type type;
      interface primary-interface-name {
         protect-interface protect-interface-name
      }
      interface protect-interface-name
   }
   route-distinguisher (as-number:number | ip-address:number);
   vrf-target community;
}
```

To configure the protect interface in a routing instance, configure both the protect interface and the primary interface in the routing instance for EVPN-VPWS and include the **protect-interface** statement in the **evpn** protocol hierarchy.

```
routing-instances {
  routing-instance-name {
    instance-type evpn-vpws;
    interface primary-interface-name ;
    interface protect-interface-name;
    route-distinguisher (as-number:number | ip-address:number);
    vrf-target community;
    protocols {
      evpn {
        interface primary-interface-name {
          vpws-service-id {
            local service-id;
            remote service-id;
            }
          protect-interface protect-interface-name
        }
      }
    }
  }
}
```

To configure the protect interface in a bridge domain, configure both the protect interface and the primary interface in the bridge domain and include the **protect-interface** statement in the primary interface hierarchy.

```
bridge-domains {
  bridge-domain-name{
    domain-type bridge;
    vlan-id number;
    interface primary-interface-name {
      protect-interface protect-interface-name
      }
    interface protect-interface-name
  }
}
```

To display the protect interface, use the **show evpn instance extensive** operational command.

```
user@PE1> show evpn instance extensive

Instance: blue
Route Distinguisher: 10.255.255.1:100
Per-instance MAC route label: 299776
MAC database status Local Remote
MAC advertisements: 0 0
MAC+IP advertisements: 0 0
Default gateway MAC advertisements: 0 0
Number of local interfaces: 5 (5 up)
Interface name ESI Mode Status AC-Role
```

```
ae0.0 00:11:22:33:44:55:66:77:88:99 all-active Up Root
ge-0/0/3.0 00:00:00:00:00:00:00:00:00:00 single-homed Up Root
ge-0/0/4.0 00:11:11:11:44:55:66:77:88:99 all-active Up Root
ge-0/0/4.1 00:22:22:22:44:55:66:77:88:99 all-active Up Root
ge-0/0/4.50 00:00:00:00:00:00:00:00:00:00 single-homed Up Root
Number of IRB interfaces: 1 (0 up)
Interface name VLAN VNI Status L3 context
irb.1 25 Down vrf
Number of protect interfaces: 1
Interface name Protect Interface name Status
ge-0/0/3.1 ge-0/0/4.50 Protect-inactive
```

The **show interfaces detail** command shows that the protect interface has a CCC-DOWN status

```
user@PE1> show interfaces ae0.0 detail
```

```
  Logical interface ae80.0 (Index 399) (SNMP ifIndex 612) (Generation 223)
    Flags: Up SNMP-Traps CCC-Down 0x20004000 VLAN-Tag [ 0x8100.10 ]  Encapsulation:
VLAN-Bridge
    Statistics        Packets        pps         Bytes          bps
    Bundle:
        Input :             0          0             0            0
        Output:             0          0             0            0
    Adaptive Statistics:
        Adaptive Adjusts:          0
        Adaptive Scans  :          0
        Adaptive Updates:          0
    Link:
      ge-0/1/8.0
        Input :             0          0             0            0
        Output:             0          0             0            0


    Aggregate member links: 1

    Marker Statistics:   Marker Rx     Resp Tx   Unknown Rx   Illegal Rx
      ge-0/1/8.0                0           0            0            0
    Protocol bridge, MTU: 1522, Generation: 263, Route table: 11, Mesh Group:
__all_ces__
```

Related Documentation

- *protect-interface*

Configuring EVPN Active-Standby Multihoming

You can configure an Ethernet VPN (EVPN) with multihoming support to provide multihoming functionality with active-standby redundancy mode of operation in the EVPN and virtual switch routing instance. This mode enables autodiscovery of Ethernet segments, Ethernet segment route construction, and Ethernet segment identifier (ESI) label assignment.

When configuring active-standby EVPN multihoming, be aware of the following limitations:

- An interface or ESI can be attached to more than one EVPN instance (EVI), with a maximum limit of 200 EVIs per ESI.
- For an EVPN routing instance, only one logical interface per physical interface or ESI can be attached to an EVI.
- For a virtual switch routing instance, only one logical interface per physical interface or ESI can be configured under a bridge domain.
- All the PE routers in the network topology should be running Junos OS Release 14.1 or later releases, which are based on the EVPN draft-ietf-l2vpn-evpn-03. Junos OS releases prior to 14.1 support the older version of the EVPN draft, causing interoperability issues when Junos OS Release 14.1 and a previous release are running.

Before you begin:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Configure OSPF or any other IGP protocol.
4. Configure a BGP internal group.
5. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
6. Configure LDP.
7. Configure MPLS.
8. Configure RSVP MPLS LSP or GRE tunnels.

To configure the PE device:

1. Enable EVPN active-standby multihoming on the multihomed interfaces.

```
[edit interfaces]
user@PE1# set interface-name vlan-tagging
user@PE1# set interface-name encapsulation flexible-ethernet-services
user@PE1# set interface-name esi esi-value
user@PE1# set interface-name esi single-active
user@PE1# set interface-name unit 0 encapsulation vlan-bridge
user@PE1# set interface-name unit 0 vlan-id VLAN-ID
```

   For example:

```
[edit interfaces]
user@PE1# set ge-0/0/4 vlan-tagging
user@PE1# set ge-0/0/4 encapsulation flexible-ethernet-services
user@PE1# set ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
user@PE1# set ge-0/0/4 esi single-active
user@PE1# set ge-0/0/4 unit 0 encapsulation vlan-bridge
user@PE1# set ge-0/0/4 unit 0 vlan-id 300
```

2. In configuration mode, go to the following hierarchy level:

```
[edit]
user@PE1# edit routing-instances
```

3. Configure the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set virtual-switch-instance instance-type virtual-switch
```

4. Configure the extended VLAN list for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set virtual-switch-instance protocols evpn extended-vlan-list VLAN-ID
```

5. Set the type for the bridging domain in the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name
  domain-type bridge
```

6. Set the VLAN identifier for the bridging domain in the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name vlan-id
  VLAN-ID
```

7. Configure the interfaces for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name interface
  interface-name
user@PE1# set virtual-switch-instance bridge-domains bridge-domain-name
  routing-interface interface-name
```

8. Configure the route distinguisher for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set virtual-switch-instance route-distinguisher route-distinguisher-value
```

9. Configure the VPN routing and forwarding (VRF) target community for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set virtual-switch-instance vrf-target vrf-target
```

10. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type evpn
```

11. Set the VLAN identifier for the bridging domain in the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vlan-id VLAN-ID
```

12. Configure the interface names for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface interface-name
user@PE1# set evpn-instance routing-interface interface-name
```

13. Configure the route distinguisher for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

14. Configure the VPN routing and forwarding (VRF) target community for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target
```

15. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
```

16. Configure the interface names for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance interface interface-name
```

17. Configure the route distinguisher for the VRF routing instance.

```
[edit routing-instances]
```

```
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
```

18. Configure the VPN routing and forwarding (VRF) target community for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-target vrf-target
user@PE1# set vrf-instance vrf-table-label
```

19. Verify and commit the configuration.

    For example:

```
[edit routing-instances]
user@PE1# set ALPHA instance-type virtual-switch
user@PE1# set ALPHA route-distinguisher 10.255.0.1:100
user@PE1# set ALPHA vrf-target target:100:100
user@PE1# set ALPHA protocols evpn extended-vlan-list 100
user@PE1# set ALPHA bridge-domains ONE domain-type bridge
user@PE1# set ALPHA bridge-domains ONE vlan-id 100
user@PE1# set ALPHA bridge-domains ONE interface ae0.0
user@PE1# set ALPHA bridge-domains ONE interface ge-0/0/2.0
user@PE1# set ALPHA bridge-domains ONE routing-interface irb.0
user@PE1# set BETA instance-type evpn
user@PE1# set BETA vlan-id 300
user@PE1# set BETA interface ge-0/0/4.0
user@PE1# set BETA interface ae1.0
user@PE1# set BETA routing-interface irb.1
user@PE1# set BETA route-distinguisher 10.255.0.1:300
user@PE1# set BETA vrf-target target:300:300
user@PE1# set DELTA instance-type vrf
user@PE1# set DELTA interface irb.0
user@PE1# set DELTA interface irb.1
user@PE1# set DELTA route-distinguisher 10.255.0.1:200
user@PE1# set DELTA vrf-target target:200:200
user@PE1# set DELTA vrf-table-label
```

```
[edit]
user@PE1# commit
commit complete
```

Related Documentation

- Example: Configuring EVPN Active-Standby Multihoming on page 79

Example: Configuring Basic EVPN Active-Standby Multihoming

This example shows how to configure Ethernet VPN (EVPN) active-standby multihoming.

- Requirements on page 62
- Overview and Topology on page 62

Requirements

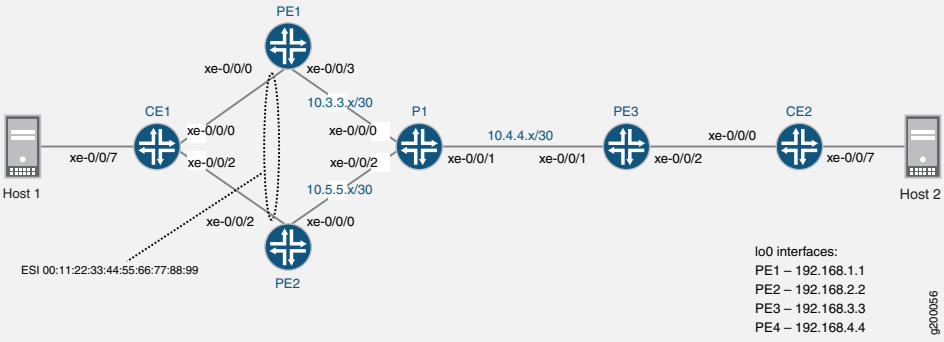
This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms running Junos OS Release 14.1 (or later), with MPC interfaces, acting as provider edge (PE) and provider (P) routers.
- Two customer edge (CE) devices.

Overview and Topology

Figure 7 on page 62 illustrates a simple EVPN topology. Routers PE1 and PE2 are provider edge (PE) routers connected to multihomed customer edge (CE) Router CE1. Router PE3 is connected to CE Router CE2.

Figure 7: Simple EVPN Multihomed Topology



The network has the following characteristics:

- All PE and P routers are running OSPF.
- There is an IBGP mesh between all PE routers.
- MPLS (RSVP) LSPs are configured between all PE routers.
- On Routers PE1 and PE2, each device's CE-facing interface uses the same Ethernet Segment Identifier (ESI).

Configuration

CLI Quick Configuration

The configurations for each device is as follows:

CE1

```
interfaces {
   xe-0/0/0 {
      description to-PE1;
      unit 0 {
```

```
        family bridge {
          interface-mode trunk;
          vlan-id-list 10;
        }
      }
    }
    xe-0/0/2 {
      description to-PE2;
      unit 0 {
        family bridge {
          interface-mode trunk;
          vlan-id-list 10;
        }
      }
    }
    xe-0/0/7 {
      description to-Host;
      unit 0 {
        family bridge {
          interface-mode access;
          vlan-id 10;
        }
      }
    }
}
bridge-domains {
    BD {
      vlan-id-list 10;
      bridge-options {
        no-mac-learning; ## Used with single-active PE configurations, ensures traffic is
          always flooded to both PEs in case of a DF change.
      }
    }
}
```

CE2

```
interfaces {
    xe-0/0/0 {
      description to-PE3;
      unit 0 {
        family bridge {
          interface-mode trunk;
          vlan-id-list 10;
        }
      }
    }
    xe-0/0/7 {
      description to-Host;
      unit 0 {
        family bridge {
          interface-mode access;
          vlan-id 10;
        }
      }
```

```
    }
}
bridge-domains {
    BD {
        vlan-id-list 10;
    }
}
```

PE1

```
interfaces {
    xe-0/0/0 {
        description to-CE1;
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        esi {
            00:11:22:33:44:55:66:77:88:99;
            single-active;
        }
        unit 10 {
            family bridge {
                interface-mode trunk;
                vlan-id-list 10;
            }
        }
    }
    xe-0/0/3 {
        description to-P;
        unit 0 {
            family inet {
                address 10.3.3.1/30;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.1.1/32;
            }
        }
    }
}
routing-options {
    router-id 192.168.1.1;
    autonomous-system 65432;
    forwarding-table {
        export evpn-pplb;
    }
}
protocols {
    rsvp {
        interface xe-0/0/3.0;
    }
    mpls {
```

```
      no-cspf;
      label-switched-path PE1-to-PE2 {
        to 192.168.2.2;
      }
      label-switched-path PE1-to-PE3 {
        to 192.168.3.3;
      }
      interface xe-0/0/3.0;
    }
    bgp {
      group EVPN-PE {
        type internal;
        local-address 192.168.1.1;
        family evpn {
          signaling;
        }
        neighbor 192.168.2.2;
        neighbor 192.168.3.3;
      }
    }
    ospf {
      area 0.0.0.0 {
        interface xe-0/0/3.0;
        interface lo0.0;
      }
    }
}
policy-options {
    policy-statement evpn-pplb {
      from protocol evpn;
      then {
        load-balance per-packet;
      }
    }
}
routing-instances {
    EVPN-RI {
      instance-type virtual-switch;
      interface xe-0/0/0.10;
      route-distinguisher 192.168.1.1:10;
      vrf-target target:65432:10;
      protocols {
        evpn {
          extended-vlan-list 10;
        }
      }
      bridge-domains {
        bd10 {
          domain-type bridge;
          vlan-id 10;
        }
      }
    }
}
```

PE2

```
interfaces {
  xe-0/0/0 {
    description to-P;
    unit 0 {
      family inet {
        address 10.5.5.1/30;
      }
      family mpls;
    }
  }
  xe-0/0/2 {
    description to-CE1;
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:11:22:33:44:55:66:77:88:99;
      single-active;
    }
    unit 10 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.2.2/32;
      }
    }
  }
}
routing-options {
  router-id 192.168.2.2;
  autonomous-system 65432;
  forwarding-table {
    export evpn-pplb;
  }
}
protocols {
  rsvp {
    interface xe-0/0/0.0;
  }
  mpls {
    no-cspf;
    label-switched-path PE2-to-PE1 {
      to 192.168.1.1;
    }
    label-switched-path PE2-to-PE3 {
      to 192.168.3.3;
    }
    interface xe-0/0/0.0;
  }
  bgp {
```

```
      group EVPN-PE {
        type internal;
        local-address 192.168.2.2;
        family evpn {
          signaling;
        }
        neighbor 192.168.1.1;
        neighbor 192.168.3.3;
      }
    }
    ospf {
      area 0.0.0.0 {
        interface xe-0/0/0.0;
        interface lo0.0;
      }
    }
}
policy-options {
    policy-statement evpn-pplb {
      from protocol evpn;
      then {
        load-balance per-packet;
      }
    }
}
routing-instances {
    EVPN-RI {
      instance-type virtual-switch;
      interface xe-0/0/2.10;
      route-distinguisher 192.168.2.2:10;
      vrf-target target:65432:10;
      protocols {
        evpn {
          extended-vlan-list 10;
        }
      }
      bridge-domains {
        bd10 {
          domain-type bridge;
          vlan-id 10;
        }
      }
    }
}
```

PE3

```
interfaces {
    xe-0/0/1 {
      description to-P;
      unit 0 {
        family inet {
          address 10.4.4.1/30;
        }
        family mpls;
```

```
    }
  }
  xe-0/0/2 {
    description to-CE3;
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.3.3/32;
      }
    }
  }
}
routing-options {
  router-id 192.168.3.3;
  autonomous-system 65432;
  forwarding-table {
    export evpn-pplb;
  }
}
protocols {
  rsvp {
    interface xe-0/0/1.0;
  }
  mpls {
    no-cspf;
    label-switched-path PE3-to-PE1 {
      to 192.168.1.1;
    }
    label-switched-path PE3-to-PE2 {
      to 192.168.2.2;
    }
    interface xe-0/0/1.0;
  }
  bgp {
    group EVPN-PE {
      type internal;
      local-address 192.168.3.3;
      family evpn {
        signaling;
      }
      neighbor 192.168.1.1;
      neighbor 192.168.2.2;
    }
  }
  ospf {
    area 0.0.0.0 {
```

```
        interface xe-0/0/1.0;
        interface lo0.0;
      }
    }
}
policy-options {
    policy-statement evpn-pplb {
      from protocol evpn;
      then {
        load-balance per-packet;
      }
    }
}
routing-instances {
    EVPN-RI {
      instance-type virtual-switch;
      interface xe-0/0/2.10;
      route-distinguisher 192.168.3.3:10;
      vrf-target target:65432:10;
      protocols {
        evpn {
          extended-vlan-list 10;
        }
      }
      bridge-domains {
        bd10 {
          domain-type bridge;
          vlan-id 10;
        }
      }
    }
}
```

P1

```
interfaces {
    xe-0/0/0 {
      unit 0 {
        family inet {
          address 10.3.3.2/30;
        }
        family mpls;
      }
    }
    xe-0/0/1 {
      unit 0 {
        family inet {
          address 10.4.4.2/30;
        }
        family mpls;
      }
    }
    xe-0/0/2 {
      unit 0 {
        family inet {
```

```
            address 10.5.5.2/30;
          }
          family mpls;
        }
      }
      lo0 {
        unit 0 {
          family inet {
            address 192.168.4.4/32;
          }
        }
      }
}
routing-options {
    router-id 192.168.4.4;
    autonomous-system 65432;
}
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface xe-0/0/0.0;
            interface xe-0/0/1.0;
            interface xe-0/0/2.0;
            interface lo0.0;
        }
    }
}
```

Verification

Confirm that the configuration is working properly.

Verifying OSPF

Purpose Verify that OSPF is working properly.

Action Verify that Router P1 has adjacencies established with all PE devices.

```
user@P1> show ospf neighbor


Address          Interface              State     ID               Pri  Dead
10.3.3.1         xe-0/0/0.0             Full      192.168.1.1      128    33
10.4.4.1         xe-0/0/1.0             Full      192.168.3.3      128    38
10.5.5.1         xe-0/0/2.0             Full      192.168.2.2      128    37
```

Meaning Adjacencies have been established with the PE devices.

Verifying BGP

Purpose Verify that BGP is working properly.

Action Verify that MP-IBGP peerings are established using EVPN signaling between all PE devices.

```
user@PE1> show bgp summary

Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                       4          4          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
192.168.2.2          65432         89         55       0       1       22:18
Establ
  EVPN-RI.evpn.0: 2/2/2/0
  bgp.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0
192.168.3.3          65432         59         48       0       1       22:18
Establ
  EVPN-RI.evpn.0: 1/1/1/0
  bgp.evpn.0: 1/1/1/0
  __default_evpn__.evpn.0: 0/0/0/0
```

```
user@PE2> show bgp summary


Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                       5          5          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
```

```
192.168.1.1           65432         80         50       0       1       22:49
Establ
  bgp.evpn.0: 4/4/4/0
  EVPN-RI.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0
192.168.3.3           65432         73         87       0       0       27:26
Establ
  bgp.evpn.0: 1/1/1/0
  EVPN-RI.evpn.0: 1/1/1/0
  __default_evpn__.evpn.0: 0/0/0/0
```

```
user@PE3> show bgp summary
```

```
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                       5          5          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
192.168.1.1           65432         66         51       0       1       23:05
Establ
  bgp.evpn.0: 3/3/3/0
  EVPN-RI.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 0/0/0/0
192.168.2.2           65432        104         64       0       0       27:42
Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-RI.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
```

Meaning EVPN-signaled MP-IBGP peerings have been established between all PE devices.

Verifying MPLS

Purpose Verify that MPLS is working properly.

Action Verify that MPLS LSPs are established between all PE devices.

```
user@PE1> show mpls lsp
```

```
Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
192.168.2.2     192.168.1.1     Up     0 *                      PE1-to-PE2
192.168.3.3     192.168.1.1     Up     0 *                      PE1-to-PE3
Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
192.168.1.1     192.168.2.2     Up       0  1 FF       3        - PE2-to-PE1
192.168.1.1     192.168.3.3     Up       0  1 FF       3        - PE3-to-PE1
Total 2 displayed, Up 2, Down 0
```

```
Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

```
user@PE2> show mpls lsp
```

```
Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
192.168.1.1     192.168.2.2     Up     0 *                      PE2-to-PE1
192.168.3.3     192.168.2.2     Up     0 *                      PE2-to-PE3
Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
192.168.2.2     192.168.3.3     Up       0  1 FF       3        - PE3-to-PE2
192.168.2.2     192.168.1.1     Up       0  1 FF       3        - PE1-to-PE2
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

```
user@PE3> show mpls lsp
```

```
Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
192.168.1.1     192.168.3.3     Up     0 *                      PE3-to-PE1
192.168.2.2     192.168.3.3     Up     0 *                      PE3-to-PE2
Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
192.168.3.3     192.168.1.1     Up       0  1 FF       3        - PE1-to-PE3
192.168.3.3     192.168.2.2     Up       0  1 FF       3        - PE2-to-PE3
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

Meaning LSPs have been established between PE devices.

Verifying EVPN Configuration and Multihoming Status

Purpose Verify that EVPN is configured properly.

Action Verify that the EVPN routing instances and ESIs are configured and functioning correctly, and confirm that single-active multihoming is enabled.

```
user@PE1> show evpn instance EVPN-RI extensive
```

```
Instance: EVPN-RI
  Route Distinguisher: 192.168.1.1:10
```

```
  Per-instance MAC route label: 300128
  MAC database status                       Local  Remote
    MAC advertisements:                         0       0
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                            Mode         Status    AC-Role

    xe-0/0/0.10     00:11:22:33:44:55:66:77:88:99  single-active   Up       Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode        MAC sync  IM route
label
    10                1    1               Extended    Enabled   300240
  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    192.168.2.2             0         0         1         1         0
    192.168.3.3             0         0         0         1         0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Status: Resolved by IFL xe-0/0/0.10
      Local interface: xe-0/0/0.10, Status: Up/Forwarding
      Number of remote PEs connected: 1
        Remote PE        MAC label  Aliasing label  Mode
        192.168.2.2      0          0               single-active
      Designated forwarder: 192.168.1.1
      Backup forwarder: 192.168.2.2
      Last designated forwarder update: Jun 26 23:30:35
      Advertised MAC label: 300224
      Advertised aliasing label: 300224
      Advertised split horizon label: 300256
```

```
user@PE2> show evpn instance EVPN-RI extensive
```

```
Instance: EVPN-RI
  Route Distinguisher: 192.168.2.2:10
  Per-instance MAC route label: 300384
  MAC database status                       Local  Remote
    MAC advertisements:                         0       0
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                            Mode        Status    AC-Role

    xe-0/0/2.10     00:11:22:33:44:55:66:77:88:99  single-active    Up       Root

  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode          MAC sync  IM route
label
    10                1    1               Extended      Enabled   300608
  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    192.168.1.1             0         0         2         1         0
    192.168.3.3             0         0         0         1         0
```

```
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Status: Resolved by NH 1048575
      Local interface: xe-0/0/2.10, Status: Up/Blocking
      Number of remote PEs connected: 1
        Remote PE        MAC label  Aliasing label    Mode
        192.168.1.1      0          300224           single-active
      Designated forwarder: 192.168.1.1
      Backup forwarder: 192.168.2.2
      Last designated forwarder update: Jun 26 23:30:43
      Advertised MAC label: 300544
      Advertised aliasing label: 300544
      Advertised split horizon label: 300320
```

```
user@PE3> show evpn instance EVPN-RI extensive
```

```
Instance: EVPN-RI
  Route Distinguisher: 192.168.3.3:10
  Per-instance MAC route label: 300272
  MAC database status                       Local  Remote
    MAC advertisements:                         0       0
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                             Mode           Status   AC-Role

    xe-0/0/2.10     00:00:00:00:00:00:00:00:00:00  single-homed   Up       Root

  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode          MAC sync  IM route
label
    10                1    1               Extended      Enabled   300368
  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    192.168.1.1             0         0         2         1         0
    192.168.2.2             0         0         1         1         0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Status: Resolved by NH 1048574
      Number of remote PEs connected: 2
        Remote PE        MAC label  Aliasing label  Mode
        192.168.1.1      0          300224          single-active
        192.168.2.2      0          0               single-active
```

Meaning From the outputs above, the following can be determined:

- All three PE devices confirm that PE1 and PE2 are using single-active mode.
- PE1 and PE2 are using the same ESI.

- PE1 is elected as the designated forwarder (DF), and its CE-facing interface is put into a state of **Up/Forwarding**.
- PE2 is elected as the backup designated forwarder (BDF), and its CE-facing interface is put into a state of **Up/Blocking**.

Verifying Route Exchange and ESI Autodiscovery

Purpose Verify that EVPN signaling is working properly.

Action Verify that autodiscovery and other signaling information is being shared between PE devices.

```
user@PE1> show route table EVPN-RI.evpn.0


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

1:192.168.1.1:10::112233445566778899::0/304 AD/EVI
                   *[EVPN/170] 00:19:27
                      Indirect
1:192.168.2.2:0::112233445566778899::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:18:20, localpref 100, from 192.168.2.2
                      AS path: I, validation-state: unverified
                    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE2
3:192.168.1.1:10::10::192.168.1.1/304 IM
                   *[EVPN/170] 00:19:31
                      Indirect
3:192.168.2.2:10::10::192.168.2.2/304 IM
                   *[BGP/170] 00:18:19, localpref 100, from 192.168.2.2
                      AS path: I, validation-state: unverified
                    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE2
3:192.168.3.3:10::10::192.168.3.3/304 IM
                   *[BGP/170] 00:18:13, localpref 100, from 192.168.3.3
                      AS path: I, validation-state: unverified
                    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE3
```

```
user@PE2> show route table EVPN-RI.evpn.0


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

1:192.168.1.1:10::112233445566778899::0/304 AD/EVI
                   *[BGP/170] 00:18:51, localpref 100, from 192.168.1.1
                      AS path: I, validation-state: unverified
                    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
1:192.168.1.1:0::112233445566778899::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:18:51, localpref 100, from 192.168.1.1
                      AS path: I, validation-state: unverified
                    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
3:192.168.1.1:10::10::192.168.1.1/304 IM
                   *[BGP/170] 00:18:51, localpref 100, from 192.168.1.1
                      AS path: I, validation-state: unverified
```

```
                    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
3:192.168.2.2:10::10::192.168.2.2/304 IM
                   *[EVPN/170] 00:18:45
                      Indirect
3:192.168.3.3:10::10::192.168.3.3/304 IM
                   *[BGP/170] 00:18:40, localpref 100, from 192.168.3.3
                      AS path: I, validation-state: unverified
                    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE3
```

user@PE3> **show route table EVPN-RI.evpn.0**

```
EVPN-RI.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:192.168.1.1:10::112233445566778899::0/304 AD/EVI
                   *[BGP/170] 00:18:54, localpref 100, from 192.168.1.1
                      AS path: I, validation-state: unverified
                    > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE1
1:192.168.1.1:0::112233445566778899::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:18:54, localpref 100, from 192.168.1.1
                      AS path: I, validation-state: unverified
                    > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE1
1:192.168.2.2:0::112233445566778899::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:18:54, localpref 100, from 192.168.2.2
                      AS path: I, validation-state: unverified
                    > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE2
3:192.168.1.1:10::10::192.168.1.1/304 IM
                   *[BGP/170] 00:18:54, localpref 100, from 192.168.1.1
                      AS path: I, validation-state: unverified
                    > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE1
3:192.168.2.2:10::10::192.168.2.2/304 IM
                   *[BGP/170] 00:18:54, localpref 100, from 192.168.2.2
                      AS path: I, validation-state: unverified
                    > to 10.4.4.2 via xe-0/0/1.0, label-switched-path PE3-to-PE2
3:192.168.3.3:10::10::192.168.3.3/304 IM
                   *[EVPN/170] 00:18:53
                      Indirect
```

Meaning The outputs above show two EVPN route types:

- Route Type 1: Ethernet Auto-Discovery (AD) Route - These routes are advertised on a per-EVI and per-ESI basis. Ethernet AD routes are required when a CE device is multihomed. When a CE device is single-homed, the ESI will be zero.
- Route Type 3: Inclusive Multicast Ethernet Tag Route - This route sets up a path for broadcast, unknown unicast, and multicast (BUM) traffic from a PE device to the remote PE device on a per VLAN, per ESI basis.

The outputs above show the following information:

- **1:192.168.x.x:10::112233445566778899::0/304 AD/EVI** - This is the per-EVI AD Type 1 EVPN route. As the DF (and active device), Router PE1 has advertised this route to Routers PE2 and PE3.

- **1:192.168.x.x:0::112233445566778899::FFFF:FFFF/304 AD/ESI** - This is the per Ethernet segment AD Type 1 EVPN route. As the multihomed devices, Routers PE1 and PE2 have advertised this route to each other and to Router PE3.
- **3:192.168.x.x:10::10::192.168.x.x/304 IM** - This is the route used to set up a path for BUM traffic. Each PE device has advertised this route to the other PE device.

Verifying Ethernet Segment (ES) Route Exchange

Purpose Verify that ES route information is being shared correctly.

Action Verify that the local and advertised autodiscovery routes per Ethernet segment and the Ethernet segment routes are received.

```
user@PE1> show route table __default_evpn__.evpn.0


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

1:192.168.1.1:0::112233445566778899::FFFF:FFFF/304 AD/ESI
                   *[EVPN/170] 00:14:22
                      Indirect
4:192.168.1.1:0::112233445566778899:192.168.1.1/304 ES
                   *[EVPN/170] 00:14:23
                      Indirect
4:192.168.2.2:0::112233445566778899:192.168.2.2/304 ES
                   *[BGP/170] 00:14:14, localpref 100, from 192.168.2.2
                      AS path: I, validation-state: unverified
                    > to 10.3.3.2 via xe-0/0/3.0, label-switched-path PE1-to-PE2
```

```
user@PE2> show route table __default_evpn__.evpn.0


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

1:192.168.2.2:0::112233445566778899::FFFF:FFFF/304 AD/ESI
                   *[EVPN/170] 00:14:25
                      Indirect
4:192.168.1.1:0::112233445566778899:192.168.1.1/304 ES
                   *[BGP/170] 00:14:24, localpref 100, from 192.168.1.1
                      AS path: I, validation-state: unverified
                    > to 10.5.5.2 via xe-0/0/0.0, label-switched-path PE2-to-PE1
4:192.168.2.2:0::112233445566778899:192.168.2.2/304 ES
                   *[EVPN/170] 00:14:26
                      Indirect
```

Meaning The outputs above show two EVPN route types:

- Route Type 1: Ethernet Auto-Discovery (AD) Route - These routes are advertised on a per-EVI and per-ESI basis. Ethernet AD routes are required when a CE device is multihomed. When a CE device is single-homed, the ESI will be zero.
- Route Type 4: Ethernet Segment Route - PE devices that are connected to the same Ethernet Segment will discover each other through the ES route.

The outputs above show the following information:

- **1:192.168.x.x:0::112233445566778899::FFFF:FFFF/304 AD/ESI** - This is the per Ethernet segment AD Type 1 EVPN route. In the outputs above, each PE device shows its own route.
- **4:192.168.x.x:0::112233445566778899:192.168.x.x/304 ES** - This is the ES route for the local ESI. In the outputs above, each PE device shows both its own route and the one advertised by the other PE device.

Related Documentation

- Example: Configuring EVPN Active-Standby Multihoming on page 79
- Example: Configuring Basic EVPN Active-Active Multihoming on page 113
- Example: Configuring EVPN Active-Active Multihoming on page 123

Example: Configuring EVPN Active-Standby Multihoming

This example shows how to configure Ethernet VPN (EVPN) for multihomed customer edge (CE) devices in an EVPN, virtual switch, and VRF routing instance, and with an integrated routing and bridging (IRB) interface configuration.

Requirements

This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms with MPC interfaces only, where:
  - Two devices are configured as provider edge (PE) routers connected to a common multihomed customer site.
  - One device is configured as a remote PE router connected to a single-homed customer site.
- Eight customer edge (CE) devices, where:
  - Two CE devices are multihomed.
  - Two CE devices are single-homed for each of the PE routers.
- Junos OS Release 14.1 or later running on all the PE routers.

NOTE: Junos OS Release 14.1 and later releases are based on the EVPN **draft-ietf-l2vpn-evpn-03**. Releases prior to 14.1, support the older version of the EVPN draft, causing interoperability issues when Junos OS Release 14.1 and a previous release are running.

Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure LDP.
5. Configure MPLS.
6. Configure RSVP MPLS LSP or GRE tunnels.

Overview and Topology

Starting with Junos OS Release 14.1, the EVPN solution on MX Series routers with MPC interfaces is extended to provide multihoming functionality with active-standby mode of operation. The multihoming functions include autodiscovery of Ethernet segments, Ethernet segment route construction, and Ethernet segment identifier (ESI) label assignment.

NOTE: Prior to Junos OS Release 15.1, the EVPN functionaliy support on MX Series Routers was limited to routers using MPC and MIC interfaces only. However, starting with Junos OS Release 15.1, MX Series Routers using DPCs can be leveraged to provide EVPN support on the CE device-facing interface.

The DPC support for EVPN is provided with the following considerations:

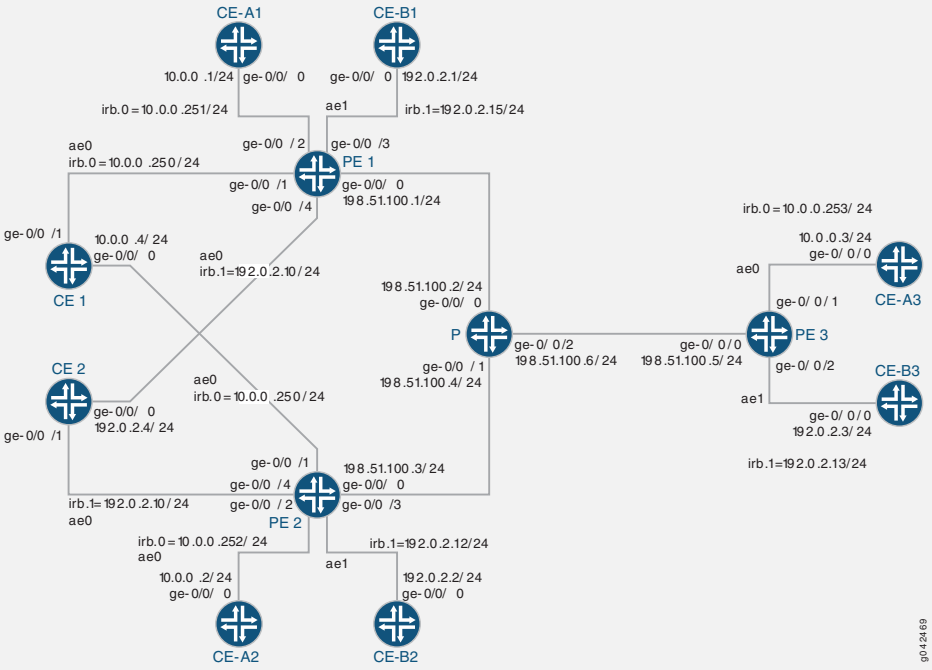
- DPCs provide support for EVPN in the active-standby mode of operation including support for the following:
  a. EVPN instance (EVI)
  b. Virtual switch (VS)
  c. Integrated routing and bridging (IRB) interfaces
- DPCs intended for providing the EVPN active-standby support should be the CE device-facing line card. The PE device interfaces in the EVPN domain should use only MPC and MIC interfaces.

NOTE:

When configuring active-standby EVPN multihoming, be aware of the following limitations:

- An interface or ESI can be attached to more than one EVI, with a maximum limit of 200 EVIs per ESI.
- For an EVPN routing instance, only one logical interface per physical interface or ESI can be attached to an EVI.
- For a virtual switch routing instance, only one logical interface per physical interface or ESI can be configured under a bridge domain.
- All the PE routers in the network topology should be running Junos OS Release 14.1 or later releases, which are based on the EVPN **draft-ietf-l2vpn-evpn-03**. Junos OS releases prior to 14.1 support the older version of the EVPN draft, causing interoperability issues when Junos OS Release 14.1 and a previous release are running.

Figure 8: EVPN Active-Standby Multihoming



In Figure 8 on page 81, Routers PE1 and PE2 are provider edge (PE) routers connected to multihomed customer edge (CE) devices, Device CE1 and CE2. Router PE3 is a remote PE router connected to a single-homed customer site, and Router P is the provider router connected to Routers PE1, PE2, and PE3.

There are three routing instances running in the topology – ALPHA, BETA, and DELTA, with the virtual switch, EVPN, and VRF type of routing instance, respectively. All the PE routers are connected to one single-homed CE device each for the ALPHA and BETA routing instances. Device CE1 belongs to the ALPHA routing instance, and Device CE2 belongs to the BETA routing instance.

For Router PE1, Device CE-A1 and Device CE-B1 are the single-homed CE devices for the routing instances ALPHA and BETA, respectively. In the same way, Device CE-A2 and Device CE-A3 belong to the ALPHA routing instance, and Device CE-B2 and Device CE-B3 belong to the BETA routing instances connected to Routers PE2 and PE3, respectively.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

CE1

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:04
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.4/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.250
```

CE-A1

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:01
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.1/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.251
```

CE-A2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:02
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.2/24
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.252
```

CE-A3

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:03
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 10.0.0.3/24
```

```
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.253
```

CE2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:04
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.4/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.10
```

CE-B1

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:01
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.1/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.15
```

CE-B2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:02
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.4/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.12
```

CE-B3

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ae0 vlan-tagging
set interfaces ae0 mac 00:00:00:00:00:03
set interfaces ae0 unit 0 vlan-id 300
set interfaces ae0 unit 0 family inet address 192.0.2.3/24
set routing-options static route 0.0.0.0/0 next-hop 192.0.2.13
```

PE1

```
set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.1/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 encapsulation flexible-ethernet-services
set interfaces ge-0/0/2 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/3 gigether-options 802.3ad ae1
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 encapsulation flexible-ethernet-services
set interfaces ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
set interfaces ge-0/0/4 esi single-active
```

```
set interfaces ge-0/0/4 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/4 unit 0 vlan-id 300
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 esi single-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae1 vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 unit 0 encapsulation vlan-bridge
set interfaces ae1 unit 0 vlan-id 300
set interfaces irb unit 0 family inet address 10.0.0.250/24
set interfaces irb unit 0 family inet address 10.0.0.251/24
set interfaces irb unit 0 mac 00:11:22:33:44:55
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 family inet address 192.0.2.15/24
set interfaces irb unit 1 mac 00:22:44:66:88:00
set interfaces lo0 unit 0 family inet address 10.255.0.1/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.1/32 preferred
set routing-options router-id 10.255.0.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols bgp group EVPN-PE type internal
set protocols bgp group EVPN-PE local-address 10.255.0.1
set protocols bgp group EVPN-PE family evpn signaling
set protocols bgp group EVPN-PE neighbor 10.255.0.2
set protocols bgp group EVPN-PE neighbor 10.255.0.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type virtual-switch
set routing-instances ALPHA route-distinguisher 10.255.0.1:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn extended-vlan-list 100
set routing-instances ALPHA bridge-domains ONE domain-type bridge
set routing-instances ALPHA bridge-domains ONE vlan-id 100
set routing-instances ALPHA bridge-domains ONE interface ae0.0
set routing-instances ALPHA bridge-domains ONE interface ge-0/0/2.0
set routing-instances ALPHA bridge-domains ONE routing-interface irb.0
set routing-instances BETA instance-type evpn
set routing-instances BETA vlan-id 300
set routing-instances BETA interface ge-0/0/4.0
set routing-instances BETA interface ae1.0
set routing-instances BETA routing-interface irb.1
set routing-instances BETA route-distinguisher 10.255.0.1:300
set routing-instances BETA vrf-target target:300:300
set routing-instances DELTA instance-type vrf
set routing-instances DELTA interface irb.0
set routing-instances DELTA interface irb.1
set routing-instances DELTA route-distinguisher 10.255.0.1:200
set routing-instances DELTA vrf-target target:200:200
```

```
set routing-instances DELTA vrf-table-label
```

PE2

```
set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.3/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 encapsulation flexible-ethernet-services
set interfaces ge-0/0/2 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/3 gigether-options 802.3ad ae1
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 encapsulation flexible-ethernet-services
set interfaces ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
set interfaces ge-0/0/4 esi single-active
set interfaces ge-0/0/4 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/4 unit 0 vlan-id 300
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 esi single-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae1 vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 unit 0 encapsulation vlan-bridge
set interfaces ae1 unit 0 vlan-id 300
set interfaces irb unit 0 family inet address 10.0.0.250/24
set interfaces irb unit 0 family inet address 10.0.0.252/24
set interfaces irb unit 0 mac 00:11:22:33:44:55
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 family inet address 192.0.2.12/24
set interfaces irb unit 1 mac 00:22:44:66:88:00
set interfaces lo0 unit 0 family inet address 10.255.0.2/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.2/32 preferred
set routing-options router-id 10.255.0.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols bgp group EVPN-PE type internal
set protocols bgp group EVPN-PE local-address 10.255.0.2
set protocols bgp group EVPN-PE family evpn signaling
set protocols bgp group EVPN-PE neighbor 10.255.0.1
set protocols bgp group EVPN-PE neighbor 10.255.0.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type virtual-switch
set routing-instances ALPHA route-distinguisher 10.255.0.2:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn extended-vlan-list 100
set routing-instances ALPHA bridge-domains ONE domain-type bridge
```

```
set routing-instances ALPHA bridge-domains ONE vlan-id 100
set routing-instances ALPHA bridge-domains ONE interface ae0.0
set routing-instances ALPHA bridge-domains ONE interface ge-0/0/2.0
set routing-instances ALPHA bridge-domains ONE routing-interface irb.0
set routing-instances BETA instance-type evpn
set routing-instances BETA vlan-id 300
set routing-instances BETA interface ge-0/0/4.0
set routing-instances BETA interface ae1.0
set routing-instances BETA routing-interface irb.1
set routing-instances BETA route-distinguisher 10.255.0.2:300
set routing-instances BETA vrf-target target:300:300
set routing-instances DELTA instance-type vrf
set routing-instances DELTA interface irb.0
set routing-instances DELTA interface irb.1
set routing-instances DELTA route-distinguisher 10.255.0.2:200
set routing-instances DELTA vrf-target target:200:200
set routing-instances DELTA vrf-table-label
```

PE3

```
set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.5/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-0/0/2 gigether-options 802.3ad ae1
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae1 vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 unit 0 encapsulation vlan-bridge
set interfaces ae1 unit 0 vlan-id 300
set interfaces irb unit 0 family inet address 10.0.0.253/24
set interfaces irb unit 1 family inet address 192.0.2.13/24
set interfaces lo0 unit 0 family inet address 10.255.0.3/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.3/32 preferred
set routing-options router-id 10.255.0.3
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols bgp group EVPN-PE type internal
set protocols bgp group EVPN-PE local-address 10.255.0.3
set protocols bgp group EVPN-PE family evpn signaling
set protocols bgp group EVPN-PE neighbor 10.255.0.1
set protocols bgp group EVPN-PE neighbor 10.255.0.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type virtual-switch
set routing-instances ALPHA route-distinguisher 10.255.0.3:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn extended-vlan-list 100
set routing-instances ALPHA bridge-domains ONE domain-type bridge
```

```
set routing-instances ALPHA bridge-domains ONE vlan-id 100
set routing-instances ALPHA bridge-domains ONE interface ae0.0
set routing-instances ALPHA bridge-domains ONE routing-interface irb.0
set routing-instances BETA instance-type evpn
set routing-instances BETA vlan-id 300
set routing-instances BETA interface ae1.0
set routing-instances BETA routing-interface irb.1
set routing-instances BETA route-distinguisher 10.255.0.3:300
set routing-instances BETA vrf-target target:300:300
set routing-instances DELTA instance-type vrf
set routing-instances DELTA interface irb.0
set routing-instances DELTA interface irb.1
set routing-instances DELTA route-distinguisher 10.255.0.3:200
set routing-instances DELTA vrf-target target:200:200
set routing-instances DELTA vrf-table-label
```

P

```
set interfaces ge-0/0/0 unit 0 family inet address 198.51.100.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.4/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 198.51.100.6/24
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.0.4/32 primary
set interfaces lo0 unit 0 family inet address 10.255.0.4/32 preferred
set routing-options router-id 10.255.0.4
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface ge-0/0/2.0
set protocols ldp interface lo0.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Router PE1 interfaces.

```
[edit interfaces]
user@PE1# set ge-0/0/0 unit 0 family inet address 198.51.100.1/24
user@PE1# set ge-0/0/0 unit 0 family mpls
user@PE1# set ge-0/0/1 gigether-options 802.3ad ae0
user@PE1# set ge-0/0/2 vlan-tagging
user@PE1# set ge-0/0/2 encapsulation flexible-ethernet-services
user@PE1# set ge-0/0/2 unit 0 encapsulation vlan-bridge
user@PE1# set ge-0/0/2 unit 0 vlan-id 100
user@PE1# set ge-0/0/3 gigether-options 802.3ad ae1
user@PE1# set ge-0/0/4 vlan-tagging
user@PE1# set ge-0/0/4 encapsulation flexible-ethernet-services
user@PE1# set ge-0/0/4 esi 00:22:44:66:88:00:22:44:66:88
user@PE1# set ge-0/0/4 esi single-active
user@PE1# set ge-0/0/4 unit 0 encapsulation vlan-bridge
user@PE1# set ge-0/0/4 unit 0 vlan-id 300
user@PE1# set ae0 vlan-tagging
user@PE1# set ae0 encapsulation flexible-ethernet-services
user@PE1# set ae0 esi 00:11:22:33:44:55:66:77:88:99
user@PE1# set ae0 esi single-active
user@PE1# set ae0 unit 0 encapsulation vlan-bridge
user@PE1# set ae0 unit 0 vlan-id 100
user@PE1# set ae1 vlan-tagging
user@PE1# set ae1 encapsulation flexible-ethernet-services
user@PE1# set ae1 unit 0 encapsulation vlan-bridge
user@PE1# set ae1 unit 0 vlan-id 300
user@PE1# set irb unit 0 family inet address 10.0.0.250/24
user@PE1# set irb unit 0 family inet address 10.0.0.251/24
user@PE1# set irb unit 0 mac 00:11:22:33:44:55
user@PE1# set irb unit 1 family inet address 192.0.2.10/24
user@PE1# set irb unit 1 family inet address 192.0.2.15/24
user@PE1# set irb unit 1 mac 00:22:44:66:88:00
user@PE1# set lo0 unit 0 family inet address 10.255.0.1/32 primary
user@PE1# set lo0 unit 0 family inet address 10.255.0.1/32 preferred
```

2. Configure the loopback address of Router PE1 as the router ID.

```
[edit routing-options]
user@PE1# set router-id 10.255.0.1
```

3. Set the autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set autonomous-system 100
```

4. Enable chained composite next hop for the EVPN.

```
[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn
```

5. Enable MPLS on the loopback interface of Router PE1 and the interface connecting PE1 to Router P.

```
[edit protocols]
user@PE1# set mpls interface lo0.0
user@PE1# set mpls interface ge-0/0/0.0
```

6. Configure the BGP group for Router PE1.

```
[edit protocols]
user@PE1# set bgp group EVPN-PE type internal
```

7. Assign local and neighbor addresses to the EVPN-PE BGP group for Router PE1 to peer with Routers PE2 and PE3.

```
[edit protocols]
user@PE1# set bgp group EVPN-PE local-address 10.255.0.1
user@PE1# set bgp group EVPN-PE neighbor 10.255.0.2
user@PE1# set bgp group EVPN-PE neighbor 10.255.0.3
```

8. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the EVPN-PE group.

```
[edit protocols]
user@PE1# set bgp group EVPN-PE family evpn signaling
```

9. Configure OSPF on the loopback interface of Router PE1 and the interface connecting PE1 to Router P.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface lo0.0 passive
user@PE1# set ospf area 0.0.0.0 interface ge-0/0/0.0
```

10. Enable LDP on the loopback interface of Router PE1 and the interface connecting PE1 to Router P.

```
[edit protocols]
user@PE1# set ldp interface lo0.0
user@PE1# set ldp interface ge-0/0/0.0
```

11. Configure the virtual switch routing instance – ALPHA.

```
[edit routing-instances]
user@PE1# set ALPHA instance-type virtual-switch
```

12. Configure the extended VLAN list for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA protocols evpn extended-vlan-list 100
```

13. Set the type for the bridging domain in the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA bridge-domains ONE domain-type bridge
```

14. Set the VLAN for the bridging domain in the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA bridge-domains ONE vlan-id 100
```

15. Configure the interface names for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA bridge-domains ONE interface ae0.0
user@PE1# set ALPHA bridge-domains ONE interface ge-0/0/2.0
user@PE1# set ALPHA bridge-domains ONE routing-interface irb.0
```

16. Configure the route distinguisher for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA route-distinguisher 10.255.0.1:100
```

17. Configure the VPN routing and forwarding (VRF) target community for the ALPHA routing instance.

```
[edit routing-instances]
user@PE1# set ALPHA vrf-target target:100:100
```

18. Configure the EVPN routing instance – BETA.

```
[edit routing-instances]
user@PE1# set BETA instance-type evpn
```

19. Set the VLAN identifier for the bridging domain in the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA vlan-id 300
```

20. Configure the interface names for the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA interface ge-0/0/4.0
```

```
user@PE1# set BETA interface ae1.0
user@PE1# set BETA routing-interface irb.1
```

21. Configure the route distinguisher for the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA route-distinguisher 10.255.0.1:300
```

22. Configure the VPN routing and forwarding (VRF) target community for the BETA routing instance.

```
[edit routing-instances]
user@PE1# set BETA vrf-target target:300:300
```

23. Configure the VRF routing instance – DELTA.

```
[edit routing-instances]
user@PE1# set DELTA instance-type vrf
```

24. Configure the interface names for the DELTA routing instance.

```
[edit routing-instances]
user@PE1# set DELTA interface irb.0
user@PE1# set DELTA interface irb.1
```

25. Configure the route distinguisher for the DELTA routing instance.

```
[edit routing-instances]
user@PE1# set DELTA route-distinguisher 10.255.0.1:200
```

26. Configure the VPN routing and forwarding (VRF) target community for the DELTA routing instance.

```
[edit routing-instances]
user@PE1# set DELTA vrf-target target:200:200
user@PE1# set DELTA vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
```

```
ge-0/0/0 {
    unit 0 {
        family inet {
            address 198.51.100.1/24;
        }
        family mpls;
    }
}
ge-0/0/1 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-0/0/2 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 100;
    }
}
ge-0/0/3 {
    gigether-options {
        802.3ad ae1;
    }
}
ge-0/0/4 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:22:44:66:88:00:22:44:66:88;
        single-active;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 300;
    }
}
ae0 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:11:22:33:44:55:66:77:88:99;
        single-active;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 100;
    }
}
ae1 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
```

```
      vlan-id 300;
    }
  }
  irb {
    unit 0 {
      family inet {
        address 10.0.0.250/24;
        address 10.0.0.251/24;
      }
      mac 00:11:22:33:44:55;
    }
    unit 1 {
      family inet {
        address 192.0.2.10/24;
        address 192.0.2.15/24;
      }
      mac 00:22:44:66:88:00;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.0.1/32 {
          primary;
          preferred;
        }
      }
    }
  }
```

```
user@PE1# show routing-options
router-id 10.255.0.1;
autonomous-system 100;
forwarding-table {
  chained-composite-next-hop {
    ingress {
      evpn;
    }
  }
}
```

```
user@PE1# show protocols
mpls {
  interface lo0.0;
  interface ge-0/0/0.0;
}
bgp {
  group EVPN-PE {
    type internal;
    local-address 10.255.0.1;
    family evpn {
      signaling;
    }
    neighbor 10.255.0.2;
```

```
      neighbor 10.255.0.3;
    }
  }
  ospf {
    area 0.0.0.0 {
      interface lo0.0 {
        passive;
      }
      interface ge-0/0/0.0;
    }
  }
  ldp {
    interface ge-0/0/0.0;
    interface lo0.0;
  }
```

```
user@PE1# show routing-instances
ALPHA {
    instance-type virtual-switch;
    route-distinguisher 10.255.0.1:100;
    vrf-target target:100:100;
    protocols {
        evpn {
            extended-vlan-list 100;
        }
    }
    bridge-domains {
        ONE {
            domain-type bridge;
            vlan-id 100;
            interface ae0.0;
            interface ge-0/0/2.0;
            routing-interface irb.0;
        }
    }
}
BETA {
    instance-type evpn;
    vlan-id 300;
    interface ge-0/0/4.0;
    interface ae1.0;
    routing-interface irb.1;
    route-distinguisher 10.255.0.1:300;
    vrf-target target:300:300;
}
DELTA {
    instance-type vrf;
    interface irb.0;
    interface irb.1;
    route-distinguisher 10.255.0.1:200;
    vrf-target target:200:200;
    vrf-table-label;
}
```

Verification

Confirm that the configuration is working properly in the following different areas on all the PE routers, where Router PE1 is the designated forwarder (DF), Router PE2 is the non-DF, and Router PE3 is the remote PE:

a. EVPN routing instance configuration

b. EVPN multihoming routes

c. DF election process

d. IRB and virtual switch routing instance configuration

e. Host route entries

Verifying the EVPN Instance Status

Purpose Verify the EVPN routing instances and their status.

Router PE1 **Action**

From operational mode, run the **show evpn instance extensive** command.

```
user@PE1> show evpn instance extensive

Instance: ALPHA
  Route Distinguisher: 10.255.0.1:100
  Per-instance MAC route label: 300144
  Per-instance multicast route label: 300160
  MAC database status                 Local  Remote
    Total MAC addresses:                 3       4
    Default gateway MAC addresses:       1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                            Mode             SH label
    ae0.0           00:11:22:33:44:55:66:77:88:99  single-active
    ge-0/0/2.0      00:00:00:00:00:00:00:00:00:00  single-homed
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.0           DELTA
```

```
  Number of neighbors: 2
    10.255.0.2
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           3
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
    10.255.0.3
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           2
        Inclusive multicast:                    1
        Ethernet auto-discovery:                0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: BETA
  Route Distinguisher: 10.255.0.1:300
  VLAN ID: 300
  Per-instance MAC route label: 300176
  Per-instance multicast route label: 300192
  MAC database status                Local  Remote
    Total MAC addresses:                 3       4
    Default gateway MAC addresses:       1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                               Mode             SH label
    ae1.0           00:00:00:00:00:00:00:00:00:00     single-homed
    ge-0/0/4.0      00:22:44:66:88:00:22:44:66:88     single-active
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.1           DELTA
  Number of neighbors: 2
    10.255.0.2
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           3
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
    10.255.0.3
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           2
        Inclusive multicast:                    1
        Ethernet auto-discovery:                0
  Number of ethernet segments: 1
    ESI: 00:22:44:66:88:00:22:44:66:88
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: __default_evpn__
  Route Distinguisher: 10.255.0.1:0
  VLAN ID: 0
  Per-instance MAC route label: 300208
  Per-instance multicast route label: 300224
  MAC database status                Local  Remote
    Total MAC addresses:                 0       0
    Default gateway MAC addresses:       0       0
```

```
  Number of local interfaces: 0 (0 up)
  Number of IRB interfaces: 0 (0 up)
  Number of neighbors: 1
    10.255.0.2
      Received routes
        Ethernet auto-discovery:                0
        Ethernet Segment:                       2
  Number of ethernet segments: 0
```

Router PE2

From operational mode, run the **show evpn instance extensive** command.

```
user@PE2> show evpn instance extensive
```

```
Instance: ALPHA
  Route Distinguisher: 10.255.0.2:100
  Per-instance MAC route label: 300208
  Per-instance multicast route label: 300224
  MAC database status                 Local  Remote
    Total MAC addresses:                  2       5
    Default gateway MAC addresses:        1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                               Mode             SH label
    ae0.0           00:11:22:33:44:55:66:77:88:99     single-active
    ge-0/0/2.0      00:00:00:00:00:00:00:00:00:00     single-homed
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.0           DELTA
  Number of neighbors: 2
    10.255.0.1
      Received routes
        MAC address advertisement:              3
        MAC+IP address advertisement:           4
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
    10.255.0.3
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           2
        Inclusive multicast:                    1
        Ethernet auto-discovery:                0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: BETA
  Route Distinguisher: 10.255.0.2:300
  VLAN ID: 300
  Per-instance MAC route label: 300240
  Per-instance multicast route label: 300256
  MAC database status                 Local  Remote
    Total MAC addresses:                  2       5
    Default gateway MAC addresses:        1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                               Mode             SH label
    ae1.0           00:00:00:00:00:00:00:00:00:00     single-homed
```

```
    ge-0/0/4.0      00:22:44:66:88:00:22:44:66:88  single-active
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.1           DELTA
  Number of neighbors: 2
    10.255.0.1
      Received routes
        MAC address advertisement:              3
        MAC+IP address advertisement:           4
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
    10.255.0.3
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           2
        Inclusive multicast:                    1
        Ethernet auto-discovery:                0
  Number of ethernet segments: 1
    ESI: 00:22:44:66:88:00:22:44:66:88
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: __default_evpn__
  Route Distinguisher: 10.255.0.2:0
  VLAN ID: 0
  Per-instance MAC route label: 300272
  Per-instance multicast route label: 300288
  MAC database status                Local  Remote
    Total MAC addresses:                 0       0
    Default gateway MAC addresses:       0       0
  Number of local interfaces: 0 (0 up)
  Number of IRB interfaces: 0 (0 up)
  Number of neighbors: 1
    10.255.0.1
      Received routes
        Ethernet auto-discovery:                0
        Ethernet Segment:                       2
  Number of ethernet segments: 0
```

Router PE3

From operational mode, run the **show evpn instance extensive** command.

```
user@PE3> show evpn instance extensive
```

```
Instance: ALPHA
  Route Distinguisher: 10.255.0.3:100
  Per-instance MAC route label: 299776
  Per-instance multicast route label: 299792
  MAC database status                Local  Remote
    Total MAC addresses:                 2       4
    Default gateway MAC addresses:       1       1
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                            Mode             SH label
    ae0.0           00:00:00:00:00:00:00:00:00:00  single-homed
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.0           DELTA
```

```
  Number of neighbors: 2
    10.255.0.1
      Received routes
        MAC address advertisement:              3
        MAC+IP address advertisement:           4
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
    10.255.0.2
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           3
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
  Number of ethernet segments: 0

Instance: BETA
  Route Distinguisher: 10.255.0.3:300
  VLAN ID: 300
  Per-instance MAC route label: 299808
  Per-instance multicast route label: 299824
  MAC database status                Local  Remote
    Total MAC addresses:                 2       4
    Default gateway MAC addresses:       1       1
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                               Mode             SH label
    ae1.0           00:00:00:00:00:00:00:00:00:00     single-homed
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.1           DELTA
  Number of neighbors: 2
    10.255.0.1
      Received routes
        MAC address advertisement:              3
        MAC+IP address advertisement:           4
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
    10.255.0.2
      Received routes
        MAC address advertisement:              2
        MAC+IP address advertisement:           3
        Inclusive multicast:                    1
        Ethernet auto-discovery:                1
  Number of ethernet segments: 0

Instance: __default_evpn__
  Route Distinguisher: 10.255.0.3:0
  VLAN ID: 0
  Per-instance MAC route label: 299840
  Per-instance multicast route label: 299856
  MAC database status                Local  Remote
    Total MAC addresses:                 0       0
    Default gateway MAC addresses:       0       0
  Number of local interfaces: 0 (0 up)
  Number of IRB interfaces: 0 (0 up)
  Number of neighbors: 0
  Number of ethernet segments: 0
```

Meaning The output provides the following information:

- List of EVPN and virtual switch routing instances.
- Mode of operation of each interface
- Neighbors of each routing instance.
- Number of different routes received from each neighbor.
- ESI attached to each routing instance.
- Number of Ethernet segments on each routing instance.
- DF election roles for each ESI in an EVI.
- VLAN ID and MAC labels for each routing instance.
- IRB interface details
- Number of default gateway MAC addresses received for the virtual switch routing instance (ALPHA).

Verifying the Autodiscovery Routes per Ethernet Segment

Purpose Verify that the autodiscovery routes per Ethernet segment are received.

Router PE1 **Action**

From operational mode, run the **show route table ALPHA.evpn.0** command.

```
user@PE1> show route table ALPHA.evpn.0

ALPHA.evpn.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.2:0::112233445566778899::0/304
                   *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.1:100::100::00:00:00:00:00:01/304
                   *[EVPN/170] 2d 23:52:22
                      Indirect
2:10.255.0.1:100::100::00:00:00:00:00:04/304
                   *[EVPN/170] 2d 23:52:24
                      Indirect
2:10.255.0.1:100::100::00:11:22:33:44:55/304
                   *[EVPN/170] 2d 23:53:32
                      Indirect
2:10.255.0.2:100::100::00:00:00:00:00:02/304
                   *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55/304
                   *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:00:03/304
```

```
                  *[BGP/170] 2d 23:39:04, localpref 100, from 10.255.0.3
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299824
2:10.255.0.3:100::100::00:05:86:71:b3:f0/304
                  *[BGP/170] 2d 23:39:24, localpref 100, from 10.255.0.3
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299824
2:10.255.0.1:100::100::00:00:00:00:00:01::10.0.0.1/304
                  *[EVPN/170] 2d 23:52:22
                     Indirect
2:10.255.0.1:100::100::00:00:00:00:00:04::10.0.0.4/304
                  *[EVPN/170] 2d 23:52:24
                     Indirect
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.250/304
                  *[EVPN/170] 2d 23:53:32
                     Indirect
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.251/304
                  *[EVPN/170] 2d 23:53:32
                     Indirect
2:10.255.0.2:100::100::00:00:00:00:00:02::10.0.0.2/304
                  *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.250/304
                  *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.252/304
                  *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:00:03::10.0.0.3/304
                  *[BGP/170] 2d 23:39:04, localpref 100, from 10.255.0.3
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299824
2:10.255.0.3:100::100::00:05:86:71:b3:f0::10.0.0.253/304
                  *[BGP/170] 2d 23:39:24, localpref 100, from 10.255.0.3
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299824
3:10.255.0.1:100::100::10.255.0.1/304
                  *[EVPN/170] 2d 23:52:33
                     Indirect
3:10.255.0.2:100::100::10.255.0.2/304
                  *[BGP/170] 2d 23:51:27, localpref 100, from 10.255.0.2
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299808
3:10.255.0.3:100::100::10.255.0.3/304
                  *[BGP/170] 2d 23:39:24, localpref 100, from 10.255.0.3
                     AS path: I, validation-state: unverified
                   > to 198.51.100.2 via ge-0/0/0.0, Push 299824
```

Router PE2

From operational mode, run the **show route table ALPHA.evpn.0** command.

```
user@PE2> show show route table ALPHA.evpn.0
```

```
ALPHA.evpn.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.1:0::112233445566778899::0/304
                   *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:01/304
                   *[BGP/170] 10:43:51, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:04/304
                   *[BGP/170] 10:45:06, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55/304
                   *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:00:02/304
                   *[EVPN/170] 10:43:48
                      Indirect
2:10.255.0.2:100::100::00:11:22:33:44:55/304
                   *[EVPN/170] 10:46:04
                      Indirect
2:10.255.0.3:100::100::00:00:00:00:00:03/304
                   *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299792
2:10.255.0.3:100::100::00:05:86:71:79:f0/304
                   *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299792
2:10.255.0.1:100::100::00:00:00:00:00:01::10.0.0.1/304
                   *[BGP/170] 10:41:52, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:04::10.0.0.4/304
                   *[BGP/170] 10:45:06, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.250/304
                   *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.251/304
                   *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:00:02::10.0.0.2/304
                   *[EVPN/170] 10:40:25
                      Indirect
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.250/304
                   *[EVPN/170] 10:46:04
                      Indirect
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.252/304
                   *[EVPN/170] 10:46:04
                      Indirect
2:10.255.0.3:100::100::00:00:00:00:00:03::10.0.0.3/304
```

```
                    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
                       AS path: I, validation-state: unverified
                     > to 198.51.100.4 via ge-0/0/0.0, Push 299792
2:10.255.0.3:100::100::00:05:86:71:79:f0::10.0.0.253/304
                    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
                       AS path: I, validation-state: unverified
                     > to 198.51.100.4 via ge-0/0/0.0, Push 299792
3:10.255.0.1:100::100::10.255.0.1/304
                    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.1
                       AS path: I, validation-state: unverified
                     > to 198.51.100.4 via ge-0/0/0.0, Push 299776
3:10.255.0.2:100::100::10.255.0.2/304
                    *[EVPN/170] 10:46:04
                       Indirect
3:10.255.0.3:100::100::10.255.0.3/304
                    *[BGP/170] 10:46:04, localpref 100, from 10.255.0.3
                       AS path: I, validation-state: unverified
                     > to 198.51.100.4 via ge-0/0/0.0, Push 299792
```

Router PE3

From operational mode, run the **show route table ALPHA.evpn.0** command.

```
user@PE3> show route table ALPHA.evpn.0

ALPHA.evpn.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.1:0::112233445566778899::0/304
                    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
                       AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
1:10.255.0.2:0::112233445566778899::0/304
                    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
                       AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.1:100::100::00:00:00:00:00:01/304
                    *[BGP/170] 10:45:21, localpref 100, from 10.255.0.1
                       AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:04/304
                    *[BGP/170] 10:46:36, localpref 100, from 10.255.0.1
                       AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55/304
                    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
                       AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:00:02/304
                    *[BGP/170] 10:45:18, localpref 100, from 10.255.0.2
                       AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55/304
                    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
                       AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:00:03/304
                    *[EVPN/170] 10:59:05
```

```
                      Indirect
2:10.255.0.3:100::100::00:05:86:71:79:f0/304
                    *[EVPN/170] 11:00:23
                      Indirect
2:10.255.0.1:100::100::00:00:00:00:00:01::10.0.0.1/304
                    *[BGP/170] 10:43:22, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:00:00:00:00:04::10.0.0.4/304
                    *[BGP/170] 10:46:36, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.250/304
                    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.1:100::100::00:11:22:33:44:55::10.0.0.251/304
                    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
2:10.255.0.2:100::100::00:00:00:00:00:02::10.0.0.2/304
                    *[BGP/170] 10:41:55, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.250/304
                    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.2:100::100::00:11:22:33:44:55::10.0.0.252/304
                    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299808
2:10.255.0.3:100::100::00:00:00:00:00:03::10.0.0.3/304
                    *[EVPN/170] 10:59:05
                      Indirect
2:10.255.0.3:100::100::00:05:86:71:79:f0::10.0.0.253/304
                    *[EVPN/170] 11:00:23
                      Indirect
3:10.255.0.1:100::100::10.255.0.1/304
                    *[BGP/170] 10:47:43, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299776
3:10.255.0.2:100::100::10.255.0.2/304
                    *[BGP/170] 10:47:34, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ge-0/0/0.0, Push 299808
3:10.255.0.3:100::100::10.255.0.3/304
                    *[EVPN/170] 10:59:40
                      Indirect
```

Meaning The remote type 1 autodiscovery route is received for the ESI attached to Router PE2, which is the other PE router connected to the multihomed CE device.

Verifying the Ethernet Segment Route

Purpose Verify that the local and advertised autodiscovery routes per Ethernet segment and the Ethernet segment routes are received.

Router PE1

Action

From operational mode, run the **show route table __default_evpn__.evpn.0** command.

```
user@PE1> show route table __default_evpn__.evpn.0

__default_evpn__.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.1:0::112233445566778899::0/304
                   *[EVPN/170] 3d 00:00:31
                      Indirect
1:10.255.0.1:0::224466880022446688::0/304
                   *[EVPN/170] 3d 00:00:31
                      Indirect
4:10.255.0.1:0::112233445566778899:10.255.0.1/304
                   *[EVPN/170] 3d 00:00:31
                      Indirect
4:10.255.0.1:0::224466880022446688:10.255.0.1/304
                   *[EVPN/170] 3d 00:00:31
                      Indirect
4:10.255.0.2:0::112233445566778899:10.255.0.2/304
                   *[BGP/170] 3d 00:00:20, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
4:10.255.0.2:0::224466880022446688:10.255.0.2/304
                   *[BGP/170] 3d 00:00:20, localpref 100, from 10.255.0.2
                      AS path: I, validation-state: unverified
                    > to 198.51.100.2 via ge-0/0/0.0, Push 299808
```

Router PE2

From operational mode, run the **show route table __default_evpn__.evpn.0** command.

```
user@PE2> show route table __default_evpn__.evpn.0

__default_evpn__.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.0.2:0::112233445566778899::0/304
                   *[EVPN/170] 10:49:26
                      Indirect
1:10.255.0.2:0::224466880022446688::0/304
                   *[EVPN/170] 10:49:26
                      Indirect
4:10.255.0.1:0::112233445566778899:10.255.0.1/304
                   *[BGP/170] 10:49:26, localpref 100, from 10.255.0.1
                      AS path: I, validation-state: unverified
                    > to 198.51.100.4 via ge-0/0/0.0, Push 299776
4:10.255.0.1:0::224466880022446688:10.255.0.1/304
```

```
                  *[BGP/170] 10:49:26, localpref 100, from 10.255.0.1
                     AS path: I, validation-state: unverified
                   > to 198.51.100.4 via ge-0/0/0.0, Push 299776
4:10.255.0.2:0::112233445566778899:10.255.0.2/304
                  *[EVPN/170] 10:49:26
                     Indirect
4:10.255.0.2:0::224466880022446688:10.255.0.2/304
                  *[EVPN/170] 10:49:26
                     Indirect
```

Meaning The output displays the local and remote type 1 (autodiscovery) and type 4 (Ethernet segment) routes:

- **1:10.255.0.1:0::112233445566778899::0/304**—Autodiscovery route per Ethernet segment for each local ESI attached to Router PE1 and Router PE2.
- **4:10.255.0.1:0::112233445566778899:10.255.0.1/304**—Ethernet segment route for each local ESI attached to Router PE1 and Router PE2.
- **4:10.255.0.2:0::112233445566778899:10.255.0.2/304**—Remote Ethernet segment route from Router PE2.

Verifying the DF Status

Purpose Confirm which PE router is the designated forwarder (DF).

Action From operational mode, run the **show evpn instance ALPHA esi *esi* designated-forwarder** command.

```
user@PE1> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99
designated-forwarder
```

```
Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1
```

Meaning Router PE1 is the DF for the ALPHA routing instance.

Verifying the BDF Status

Purpose Confirm which PE router is the backup designated forwarder (BDF).

Action From operational mode, run the **show evpn instance ALPHA esi *esi* backup-forwarder** command.

```
user@PE1> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99
backup-forwarder
```

```
Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Backup forwarder: 10.255.0.2
```

Meaning Router PE2 is the BDF for the ALPHA routing instance.

Verifying Remote IRB MAC

Purpose Verify that the remote gateway MAC addresses are synchronized among all the PE routers.

Router PE1

Action

From operational mode, run the **show bridge evpn peer-gateway-mac** command.

```
user@PE1> show bridge evpn peer-gateway-mac
```

```
Routing instance : ALPHA
 Bridging domain : ONE, VLAN : 100
  Installed GW MAC addresses:
  00:05:86:71:79:f0
```

Router PE2

From operational mode, run the **show bridge evpn peer-gateway-mac** command.

```
user@PE2> show bridge evpn peer-gateway-mac
```

```
Routing instance : ALPHA
 Bridging domain : ONE, VLAN : 100
  Installed GW MAC addresses:
  00:05:86:71:79:f0
```

Router PE3

From operational mode, run the **show bridge evpn peer-gateway-mac** command.

```
user@PE2> show bridge evpn peer-gateway-mac
```

```
Routing instance : ALPHA
 Bridging domain : ONE, VLAN : 100
  Installed GW MAC addresses:
  00:11:22:33:44:55
```

Meaning The remote gateway MAC addresses are synchronized:

- Router PE3 gateway MAC is installed in Routers PE1 and PE2 peer-gateway-mac table.
- Routers PE1 and PE2 gateway MAC addresses are installed in Router PE3 peer-gateway-mac table.

Verifying Remote IRB and Host IP

Purpose Verify that the remote IRB IP and the host IP are received.

Router PE1

Action

From operational mode, run the **show route table DELTA** command.

```
user@PE1> show route table DELTA

DELTA.inet.0: 16 destinations, 18 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/24             *[Direct/0] 11:25:54
                         > via irb.0
                         [Direct/0] 11:25:54
                         > via irb.0
10.0.0.1/32             *[EVPN/7] 11:21:33
                         > via irb.0
10.0.0.2/32             *[EVPN/7] 11:20:06, metric2 1
                         > to 198.51.100.2 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.3/32             *[EVPN/7] 11:25:54, metric2 1
                         > to 198.51.100.2 via ge-0/0/0.0, Push 299776, Push 299792(top)
10.0.0.4/32              *[EVPN/7] 11:24:47
                         > via irb.0
10.0.0.250/32           *[Local/0] 11:38:29
                            Local via irb.0
10.0.0.251/32            *[Local/0] 11:38:29
                            Local via irb.0
10.0.0.253/32           *[EVPN/7] 11:25:54, metric2 1
                         > to 198.51.100.2 via ge-0/0/0.0, Push 299776, Push 299792(top)
192.0.2.0/24             *[Direct/0] 11:25:55
                         > via irb.1
                         [Direct/0] 11:25:55
                         > via irb.1
192.0.2.1/24             *[EVPN/7] 11:21:20
                         > via irb.1
192.0.2.2/24             *[EVPN/7] 11:19:54, metric2 1
                        > to 198.51.100.2 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.3/24             *[EVPN/7] 11:25:54, metric2 1
                        > to 198.51.100.2 via ge-0/0/0.0, Push 299808, Push 299792(top)
192.0.2.4/24             *[EVPN/7] 11:24:40
                         > via irb.1
192.0.2.10/24            *[Local/0] 11:38:29
                            Local via irb.1
192.0.2.15/24            *[Local/0] 11:38:29
                            Local via irb.1
```

```
192.0.2.13/24      *[EVPN/7] 11:25:54, metric2 1
                  > to 198.51.100.2 via ge-0/0/0.0, Push 299808, Push 299792(top)
```

Router PE2

From operational mode, run the **show route table DELTA** command.

```
user@PE2> show route table DELTA
```

```
DELTA.inet.0: 16 destinations, 18 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/24        *[Direct/0] 11:30:02
                    > via irb.0
                    [Direct/0] 11:30:02
                    > via irb.0
10.0.0.1/32        *[EVPN/7] 11:25:50, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 300144, Push 299776(top)
10.0.0.2/32        *[EVPN/7] 11:24:23
                    > via irb.0
10.0.0.3/32        *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299776, Push 299792(top)
10.0.0.4/32        *[EVPN/7] 11:29:04, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 300144, Push 299776(top)
10.0.0.250/32      *[Local/0] 11:42:33
                      Local via irb.0
10.0.0.252/32      *[Local/0] 11:42:33
                      Local via irb.0
10.0.0.253/32      *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299776, Push 299792(top)
192.0.2.0/24        *[Direct/0] 11:30:02
                    > via irb.1
                    [Direct/0] 11:30:02
                    > via irb.1
192.0.2.1/24        *[EVPN/7] 11:25:37, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.2/24        *[EVPN/7] 11:24:11
                    > via irb.1
192.0.2.3/24       *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299808, Push 299792(top)
192.0.2.4/24        *[EVPN/7] 11:28:57, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.10/24      *[Local/0] 11:42:33
                      Local via irb.1
192.0.2.12/24      *[Local/0] 11:42:33
                      Local via irb.1
192.0.2.13/24      *[EVPN/7] 11:30:02, metric2 1
                  > to 198.51.100.4 via ge-0/0/0.0, Push 299808, Push 299792(top)
```

Router PE3

From operational mode, run the **show route table DELTA** command.

```
user@PE3> show route table DELTA
```

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

10.0.0.0/24        *[Direct/0] 11:42:15
                    > via irb.0
10.0.0.1/32        *[EVPN/7] 11:25:56, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300144, Push 299776(top)
10.0.0.2/32        *[EVPN/7] 11:24:29, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.3/32        *[EVPN/7] 11:41:39
                    > via irb.0
10.0.0.4/32        *[EVPN/7] 11:29:10, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300144, Push 299776(top)
10.0.0.250/32      *[EVPN/7] 11:30:08, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.252/32      *[EVPN/7] 11:30:08, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300208, Push 299808(top)
10.0.0.253/32      *[Local/0] 11:42:57
                      Local via irb.0
192.0.2.0/24        *[Direct/0] 11:42:15
                    > via irb.1
192.0.2.1/24        *[EVPN/7] 11:25:43, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.2/24        *[EVPN/7] 11:24:17, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.3/24       *[EVPN/7] 11:42:04
                    > via irb.1
192.0.2.4/24        *[EVPN/7] 11:29:03, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300176, Push 299776(top)
192.0.2.10/24      *[EVPN/7] 11:30:08, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.12/24      *[EVPN/7] 11:30:08, metric2 1
                   > to 198.51.100.6 via ge-0/0/0.0, Push 300240, Push 299808(top)
192.0.2.13/24      *[Local/0] 11:42:57
                      Local via irb.1
```

Meaning The output displays the local and remote IRB interfaces. It also displays the local and remote hosts that are installed in the VRF table:

On Router PE1:

- **10.0.0.1/32**—Local host in the virtual switch routing instance.
- **10.0.0.2/32** and **10.0.0.3/32**—Remote host in the virtual switch routing instance.
- **10.0.0.250/32**—Local IRB in the virtual switch routing instance.
- **10.0.0.253/32**—Remote IRB in the virtual switch routing instance.

Verifying ARP Table

Purpose Verify the ARP table entries.

Router PE1 **Action**

From operational mode, run the **show evpn arp-table** command.

```
user@PE1> show evpn arp-table
```

```
INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
192.0.2.1           00:00:00:00:00:01  irb.1         BETA         __BETA__
192.0.2.4           00:00:00:00:00:04  irb.1         BETA         __BETA__
```

Router PE2

From operational mode, run the **show evpn arp-table** command.

```
user@PE2> show evpn arp-table
```

```
INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
192.0.2.2           00:00:00:00:00:02  irb.1         BETA         __BETA__
```

Router PE3

From operational mode, run the **show evpn arp-table** command.

```
user@PE3> show evpn arp-table
```

```
INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
192.0.2.3           00:00:00:00:00:03  irb.1         BETA         __BETA__
```

Meaning The EVPN instance and ARP are synchronized with the host MAC and IP address for local hosts.

Verifying Bridge ARP Table

Purpose Verify the bridge ARP table entries.

Router PE1 **Action**

From operational mode, run the **show bridge evpn arp-table** command.

```
user@PE3> show bridge evpn arp-table
```

```
INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
10.0.0.1           00:00:00:00:00:01  irb.0         ALPHA        ONE
10.0.0.4           00:00:00:00:00:04  irb.0         ALPHA        ONE
```

Router PE2

From operational mode, run the **show bridge evpn arp-table** command.

```
user@PE3> show bridge evpn arp-table

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
10.0.0.2           00:00:00:00:00:02 irb.0         ALPHA        ONE
```

Router PE3

From operational mode, run the **show bridge evpn arp-table** command.

```
user@PE3> show bridge evpn arp-table

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
10.0.0.3           00:00:00:00:00:03 irb.0         ALPHA        ONE
```

Meaning The virtual switch instance and ARP are synchronized with the local host MAC and IP address.

Verifying Bridge MAC Table

Purpose Verify the bridge MAC table entries.

Router PE1 **Action**

From operational mode, run the **show bridge mac-table** command.

```
user@PE1> show bridge mac-table

MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : ALPHA
 Bridging domain : ONE, VLAN : 100
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:00:00:00:00:01   D        ge-0/0/2.0
   00:00:00:00:00:02   DC                        1048583 1048583
   00:00:00:00:00:03   DC                        1048574 1048574
   00:00:00:00:00:04   D        ae0.0
```

Router PE2

From operational mode, run the **show bridge mac-table** command.

```
user@PE2> show bridge mac-table
```

```
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : ALPHA
 Bridging domain : ONE, VLAN : 100
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:00:00:00:00:01   DC                        1048577 1048577
   00:00:00:00:00:02   D        ge-0/0/2.0
   00:00:00:00:00:03   DC                        1048578 1048578
   00:00:00:00:00:04   DC                        1048577 1048577
```

Router PE3

From operational mode, run the **show bridge mac-table** command.

```
user@PE3> show bridge mac-table
```

```
MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : ALPHA
 Bridging domain : ONE, VLAN : 100
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:00:00:00:00:01   DC                        1048575 1048575
   00:00:00:00:00:02   DC                        1048582 1048582
   00:00:00:00:00:03   D        ae0.0
   00:00:00:00:00:04   DC                        1048575 1048575
```

Meaning The virtual switch instance installed local and remote host MAC addresses in the bridge MAC table.

Related Documentation

- EVPN Multihoming Overview on page 29

Example: Configuring Basic EVPN Active-Active Multihoming

This example shows how to configure an active-active multihomed customer edge (CE) devices and provider edge (PE) devices in an Ethernet VPN (EVPN).

Requirements

This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms with MPC interfaces only, where:

- Two devices are configured as provider edge (PE) routers connected to a common multihomed customer site.
- One device is configured as a remote PE router connected to a single-homed customer site.
- One device is configured as a provider router.

- · Two customer edge (CE) devices, where:
  - Two CE devices that are multihomed.
  - One CE devices that is single-homed .

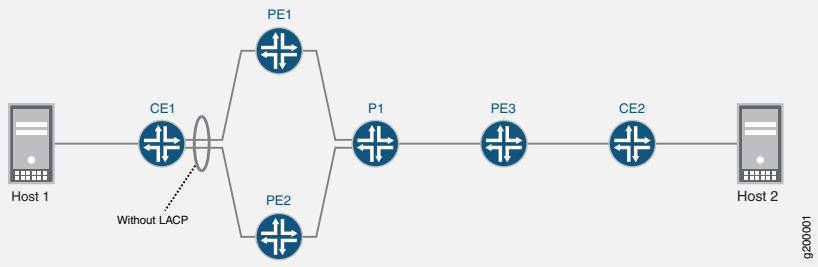
Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure a BGP internal group.
4. Configure MPLS.

Overview

Figure 7 on page 62 illustrates a simple EVPN topology. Routers PE1 and PE2 are provider edge (PE) routers connected to multihomed customer edge (CE) router CE1. Router P is the provider router connected to Routers PE1, PE2, and PE3. Router PE3 is connected to customer edge router CE2.

Figure 9: Simple EVPN Multihomed Topology



Configuration

CLI Quick Configuration

The configurations parameters for the routers are as follows:

CE1

```
chassis {
   aggregated-devices {
      ethernet {
         device-count 1;
      }
   }
```

```
}
interfaces {
    ge-0/0/1 {
        description To-PE1;
        gigether-options {
            802.3ad ae0;
        }
    }
    ge-0/0/2 {
        description To-PE2;
        gigether-options {
            802.3ad ae0;
        }
    }
    ge-0/0/7 {
        unit 0 {
            family bridge {
                interface-mode access;
                vlan-id 10;
            }
        }
    }
    ge-0/0/8 {
        unit 0 {
            family bridge {
                interface-mode access;
                vlan-id 20;
            }
        }
    }
    ae0 {
        description To-PE1_and_PE2;
        unit 0 {
            family bridge {
                interface-mode trunk;
                vlan-id-list [ 10 20 ];
            }
        }
    }
}
bridge-domains {
    BD {
        vlan-id-list [ 10 20 ];
    }
}
```

PE1

```
interfaces {
    ge-0/0/1 {
        description To-CE1;
        flexible-vlan-tagging;
        encapsulation flexible-ethernet-services;
        esi {
            00:11:11:11:11:11:11:11:11:11;
```

```
        all-active;
      }
      unit 10 {
        family bridge {
          interface-mode trunk;
          vlan-id-list [ 10 20 ];
        }
      }
    }
    ge-0/0/3 {
      unit 0 {
        family inet {
          address 10.3.3.1/30;
          }
        family mpls;
      }
    }
    lo0 {
      unit 0 {
        family inet {
          address 192.168.1.1/32;
        }
      }
    }
}
routing-options {
    router-id 192.168.1.1;
    autonomous-system 65432;
    forwarding-table {
      export evpn-pplb;
    }
}
protocols {
    rsvp {
      interface ge-0/0/3.0;
    }
    mpls {
      no-cspf;
      label-switched-path PE1-to-PE2 {
        to 192.168.2.2;
      }
      label-switched-path PE1-to-PE3 {
        to 192.168.3.3;
      }
      interface ge-0/0/3.0;
    }
    bgp {
      group EVPN-PE {
        type internal;
        local-address 192.168.1.1;
        family inet-vpn {
          unicast;
        }
        family evpn {
          signaling;
```

```
        }
        neighbor 192.168.3.3;
        neighbor 192.168.2.2;
      }
    }
    ospf {
      traffic-engineering;
      area 0.0.0.0 {
        interface all {
          interface-type p2p;
        }
        interface fxp0.0 {
          disable;
        }
      }
    }
}
routing-instances {
    evpn-ta {
      instance-type virtual-switch;
      interface ge-0/0/1.10;
      route-distinguisher 65432:10;
      vrf-target target:65432:10;
      protocols {
        evpn {
          extended-vlan-list [ 10 20 ];
        }
      }
      bridge-domains {
        bd10 {
          domain-type bridge;
          vlan-id 10;
        }
        bd20 {
          domain-type bridge;
          vlan-id 20;
        }
      }
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
      load-balance per-packet;
    }
}
```

PE2

```
interfaces {
    ge-0/0/2 {
      description To-CE1;
      flexible-vlan-tagging;
      encapsulation flexible-ethernet-services;
      esi {
```

```
        00:11:11:11:11:11:11:11:11:11;
        all-active;
      }
      unit 10 {
        family bridge {
          interface-mode trunk;
          vlan-id-list [ 10 20 ];
        }
      }
    }
    ge-0/0/4 {
      unit 0 {
        family inet {
          address 10.4.4.1/30;
          }
        family mpls;
      }
    }
    lo0 {
      unit 0 {
        family inet {
          address 192.168.2.2/32;
        }
      }
    }
}
routing-options {
    router-id 192.168.2.2;
    autonomous-system 65432;
    forwarding-table {
      export evpn-pplb;
    }
}
protocols {
    rsvp {
      interface ge-0/0/4.0;
    }
    mpls {
      no-cspf;
      label-switched-path PE2-to-PE1 {
        to 192.168.1.1;
      }
      label-switched-path PE2-to-PE3 {
        to 192.168.3.3;
      }
      interface ge-0/0/4.0;
    }
    bgp {
      group EVPN-PE {
        type internal;
        local-address 192.168.2.2;
        family inet-vpn {
          unicast;
        }
        family evpn {
```

```
            signaling;
        }
        neighbor 192.168.1.1;
        neighbor 192.168.3.3;
      }
    }
    ospf {
      traffic-engineering;
      area 0.0.0.0 {
        interface all {
          interface-type p2p;
        }
        interface fxp0.0 {
          disable;
        }
      }
    }
}
routing-instances {
    evpn-ta {
      instance-type virtual-switch;
      interface ge-0/0/2.10;
      route-distinguisher 65432:10;
      vrf-target target:65432:10;
      protocols {
        evpn {
          extended-vlan-list [ 10 20 ];
        }
      }
      bridge-domains {
        bd10 {
          domain-type bridge;
          vlan-id 10;
        }
        bd20 {
          domain-type bridge;
          vlan-id 20;
        }
      }
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
      load-balance per-packet;
    }
}
```

Router PE3

```
interfaces {
    ge-0/0/5 {
      unit 0 {
        family inet {
          address 10.5.5.1/30;
```

```
        }
      family mpls;
    }
  }
  ge-0/0/6 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 10;
      }
    }
    unit 20 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 20;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.3.3/32;
      }
    }
  }
  routing-options {
    router-id 192.168.3.3;
    autonomous-system 65432;
    forwarding-table {
      export evpn-pplb;
    }
  }
  protocols {
    rsvp {
      interface ge-0/0/5.0;
    }
    mpls {
      no-cspf;
      label-switched-path PE3-to-PE1 {
        to 192.168.1.1;
      }
      label-switched-path PE3-to-PE2 {
        to 192.168.2.2;
      }
      interface ge-0/0/5.0;
    }
    bgp {
      group EVPN-PE {
        type internal;
        local-address 192.168.3.3;
        family inet-vpn {
          unicast;
        }
```

```
            family evpn {
              signaling;
            }
            neighbor 192.168.1.1;
            neighbor 192.168.2.2;
          }
        }
        ospf {
          traffic-engineering;
          area 0.0.0.0 {
            interface all {
              interface-type p2p;
            }
            interface fxp0.0 {
              disable;
            }
          }
        }
      }
      routing-instances {
        evpn-ta {
          instance-type virtual-switch;
          interface ge-0/0/6.10;
          interface ge-0/0/6.20;
          route-distinguisher 65432:10;
          vrf-target target:65432:10;
          protocols {
            evpn {
              extended-vlan-list [ 10 20 ];
            }
          }
          bridge-domains {
            bd10 {
              domain-type bridge;
              vlan-id 10;
              bridge-options {
                interface ge-0/0/6.10;
              }
            }
            bd20 {
              domain-type bridge;
              vlan-id 20;
              bridge-options {
                interface ge-0/0/6.20;
              }
            }
          }
        }
      }
    }
    policy-statement evpn-pplb {
      from protocol evpn;
      then {
        load-balance per-packet;
      }
```

```
}
```

Router P

```
interfaces {
    ge-0/0/3 {
        unit 0 {
            family inet {
                address 10.3.3.2/30;
                }
            family mpls;
        }
    }
    ge-0/0/4 {
        unit 0 {
            family inet {
                address 10.4.4.2/30;
                }
            family mpls;
        }
    }
    ge-0/0/5 {
        unit 0 {
            family inet {
                address 10.5.5.2/30;
                }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.4.4/32;
            }
        }
    }
}
routing-options {
    router-id 192.168.4.4;
    autonomous-system 65432;
}
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    ospf {
```

```
    traffic-engineering;
    area 0.0.0.0 {
      interface all {
        interface-type p2p;
      }
      interface fxp0.0 {
        disable;
      }
    }
  }
}
```

Related Documentation

-

Example: Configuring EVPN Active-Active Multihoming

This example shows how to configure Ethernet VPN (EVPN) for multihomed customer edge devices in the active-active redundancy mode, so the Layer 2 unicast traffic can be load-balanced across all the multihomed links on and toward the CE device.

Requirements

This example uses the following hardware and software components:

- Five MX Series 5G Universal Routing Platforms with MPC interfaces only, where:
  - Three devices are configured as provider edge (PE) routers connected to a common multihomed customer site.
  - One device is configured as a remote PE router connected to a single-homed customer site.
- Six customer edge (CE) devices, with one multihomed CE device and the rest of the CE devices being single-homed to each of the PE routers.
- Junos OS Release 16.1 or later running on all the PE routers.

NOTE: The EVPN multihoming active-active mode of operation is supported in Junos OS Releases 16.1 and later releases.

Starting with Junos OS Release 16.1R4, EVPN multihoming active-active mode is supported on all EX9200 switches. For information about configuration specific to EX9200 switches, see "Configuration on EX9200 Switches" on page 147.

Before you begin:

1. Configure the router interfaces.
2. Configure IS-IS or any other IGP protocol.
3. Configure BGP.
4. Configure LDP.
5. Configure MPLS.
6. Configure RSVP MPLS LSP or GRE tunnels.

Overview and Topology

Starting with Junos OS Release 15.1, the EVPN solution on MX Series routers with MPC interfaces is extended to provide multihoming functionality in the active-active redundancy mode of operation. This feature enables load balancing of Layer 2 unicast traffic across all the multihomed links on and toward a customer edge device.

The EVPN active-active multihoming feature provides link-level and node-level redundancy along with effective utilization of resources.

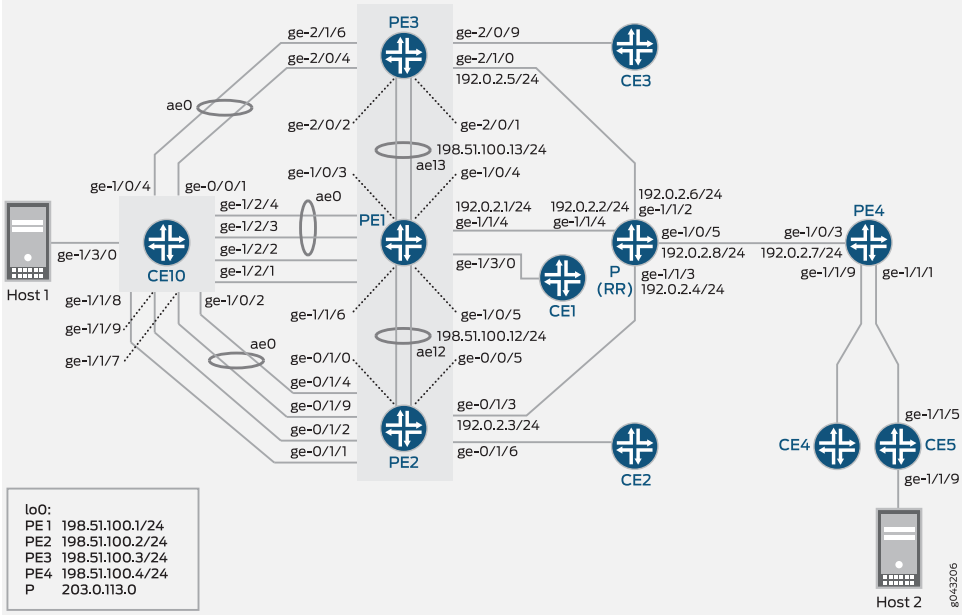
To enable EVPN active-active multihoming, include the **all-active** statement at the **[edit interfaces esi]** hierarchy level.

In Figure 10 on page 125, the core consists of four provider edge (PE) routers and a provider router (P) that is configured as a route reflector (RR). Router CE10 is multihomed to Routers PE1, PE2, and PE3. Each PE router is also connected to a single-homed customer site.

There are three routing instances running in the topology – VS-1, VS-2, and mhevpn, along with the default routing instance. The routing instances share nine VLANs with three ESIs each. The VS-1 and VS-2 routing instances are configured as a virtual switch type of routing instance, and the mhevpn routing instance is an EVPN routing instance.

Three aggregated bundles – ae0, ae1, and ae2 – are used to connect the multihomed CE device, CE10, to Routers PE1, PE2, and PE3. These aggregated bundles are configured for three ESIs each. The aggregated bundles, ae12 and ae13, are used to interconnect Routers PE1 and PE2, and PE1 and PE3, respectively.

Figure 10: EVPN Active-Active Multihoming Topology



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

CE10

```
set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-1/0/2 gigether-options 802.3ad ae0
set interfaces ge-1/0/4 gigether-options 802.3ad ae0
set interfaces ge-1/1/7 gigether-options 802.3ad ae0
set interfaces ge-1/1/8 gigether-options 802.3ad ae2
set interfaces ge-1/1/9 gigether-options 802.3ad ae1
set interfaces ge-1/2/1 gigether-options 802.3ad ae2
set interfaces ge-1/2/2 gigether-options 802.3ad ae1
set interfaces ge-1/2/3 gigether-options 802.3ad ae0
set interfaces ge-1/2/4 gigether-options 802.3ad ae0
set interfaces ge-1/3/0 flexible-vlan-tagging
set interfaces ge-1/3/0 encapsulation flexible-ethernet-services
set interfaces ge-1/3/0 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 0 vlan-id 10
set interfaces ge-1/3/0 unit 1 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 1 vlan-id 20
set interfaces ge-1/3/0 unit 2 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 2 vlan-id 30
set interfaces ge-1/3/0 unit 110 encapsulation vlan-bridge
```

```
set interfaces ge-1/3/0 unit 110 vlan-id 110
set interfaces ge-1/3/0 unit 120 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 120 vlan-id 120
set interfaces ge-1/3/0 unit 130 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 130 vlan-id 130
set interfaces ge-1/3/0 unit 210 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 210 vlan-id 210
set interfaces ge-1/3/0 unit 220 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 220 vlan-id 220
set interfaces ge-1/3/0 unit 230 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 230 vlan-id 230
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 encapsulation vlan-bridge
set interfaces ae0 unit 110 vlan-id 110
set interfaces ae0 unit 210 encapsulation vlan-bridge
set interfaces ae0 unit 210 vlan-id 210
set interfaces ae1 flexible-vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options minimum-links 1
set interfaces ae1 unit 1 encapsulation vlan-bridge
set interfaces ae1 unit 1 vlan-id 20
set interfaces ae1 unit 120 encapsulation vlan-bridge
set interfaces ae1 unit 120 vlan-id 120
set interfaces ae1 unit 220 encapsulation vlan-bridge
set interfaces ae1 unit 220 vlan-id 220
set interfaces ae2 flexible-vlan-tagging
set interfaces ae2 encapsulation flexible-ethernet-services
set interfaces ae2 aggregated-ether-options minimum-links 1
set interfaces ae2 unit 2 encapsulation vlan-bridge
set interfaces ae2 unit 2 vlan-id 30
set interfaces ae2 unit 130 encapsulation vlan-bridge
set interfaces ae2 unit 130 vlan-id 130
set interfaces ae2 unit 230 encapsulation vlan-bridge
set interfaces ae2 unit 230 vlan-id 230
set routing-options forwarding-table export load-balancing-policy
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ae0.0
set bridge-domains bd10 interface ge-1/3/0.0
set bridge-domains bd110 domain-type bridge
set bridge-domains bd110 vlan-id 110
set bridge-domains bd110 interface ae0.110
set bridge-domains bd110 interface ge-1/3/0.110
set bridge-domains bd120 domain-type bridge
set bridge-domains bd120 vlan-id 120
set bridge-domains bd120 interface ge-1/3/0.120
set bridge-domains bd120 interface ae1.120
set bridge-domains bd130 domain-type bridge
set bridge-domains bd130 vlan-id 130
```

```
set bridge-domains bd130 interface ge-1/3/0.130
set bridge-domains bd130 interface ae2.130
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface ge-1/3/0.1
set bridge-domains bd20 interface ae1.1
set bridge-domains bd210 domain-type bridge
set bridge-domains bd210 vlan-id 210
set bridge-domains bd210 interface ae0.210
set bridge-domains bd210 interface ge-1/3/0.210
set bridge-domains bd220 domain-type bridge
set bridge-domains bd220 vlan-id 220
set bridge-domains bd220 interface ge-1/3/0.220
set bridge-domains bd220 interface ae1.220
set bridge-domains bd230 domain-type bridge
set bridge-domains bd230 vlan-id 230
set bridge-domains bd230 interface ge-1/3/0.230
set bridge-domains bd230 interface ae2.230
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface ge-1/3/0.2
set bridge-domains bd30 interface ae2.2
```

CE5

```
set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-1/1/5 gigether-options 802.3ad ae0
set interfaces ge-1/1/9 flexible-vlan-tagging
set interfaces ge-1/1/9 encapsulation flexible-ethernet-services
set interfaces ge-1/1/9 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 0 vlan-id 10
set interfaces ge-1/1/9 unit 1 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 1 vlan-id 20
set interfaces ge-1/1/9 unit 2 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 2 vlan-id 30
set interfaces ge-1/1/9 unit 110 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 110 vlan-id 110
set interfaces ge-1/1/9 unit 120 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 120 vlan-id 120
set interfaces ge-1/1/9 unit 130 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 130 vlan-id 130
set interfaces ge-1/1/9 unit 210 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 210 vlan-id 210
set interfaces ge-1/1/9 unit 220 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 220 vlan-id 220
set interfaces ge-1/1/9 unit 230 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 230 vlan-id 230
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 20
set interfaces ae0 unit 2 encapsulation vlan-bridge
```

```
set interfaces ae0 unit 2 vlan-id 30
set interfaces ae0 unit 110 encapsulation vlan-bridge
set interfaces ae0 unit 110 vlan-id 110
set interfaces ae0 unit 120 encapsulation vlan-bridge
set interfaces ae0 unit 120 vlan-id 120
set interfaces ae0 unit 130 encapsulation vlan-bridge
set interfaces ae0 unit 130 vlan-id 130
set interfaces ae0 unit 210 encapsulation vlan-bridge
set interfaces ae0 unit 210 vlan-id 210
set interfaces ae0 unit 220 encapsulation vlan-bridge
set interfaces ae0 unit 220 vlan-id 220
set interfaces ae0 unit 230 encapsulation vlan-bridge
set interfaces ae0 unit 230 vlan-id 230
set interfaces lo0 unit 6 family inet address 203.0.113.0/24
set interfaces lo0 unit 6 family iso address
    47.0005.80ff.f800.0000.0108.0000.6006.0070.0600
set routing-options forwarding-table export load-balancing-policy
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ae0.0
set bridge-domains bd10 interface ge-1/1/9.0
set bridge-domains bd110 domain-type bridge
set bridge-domains bd110 vlan-id 110
set bridge-domains bd110 interface ae0.110
set bridge-domains bd110 interface ge-1/1/9.110
set bridge-domains bd120 domain-type bridge
set bridge-domains bd120 vlan-id 120
set bridge-domains bd120 interface ge-1/1/9.120
set bridge-domains bd120 interface ae0.120
set bridge-domains bd130 domain-type bridge
set bridge-domains bd130 vlan-id 130
set bridge-domains bd130 interface ge-1/1/9.130
set bridge-domains bd130 interface ae0.130
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface ae0.1
set bridge-domains bd20 interface ge-1/1/9.1
set bridge-domains bd210 domain-type bridge
set bridge-domains bd210 vlan-id 210
set bridge-domains bd210 interface ae0.210
set bridge-domains bd210 interface ge-1/1/9.210
set bridge-domains bd220 domain-type bridge
set bridge-domains bd220 vlan-id 220
set bridge-domains bd220 interface ge-1/1/9.220
set bridge-domains bd220 interface ae0.220
set bridge-domains bd230 domain-type bridge
set bridge-domains bd230 vlan-id 230
set bridge-domains bd230 interface ge-1/1/9.230
set bridge-domains bd230 interface ae0.230
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface ge-1/1/9.2
set bridge-domains bd30 interface ae0.2
```

PE1

```
set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-1/0/3 gigether-options 802.3ad ae13
set interfaces ge-1/0/4 gigether-options 802.3ad ae13
set interfaces ge-1/0/5 gigether-options 802.3ad ae12
set interfaces ge-1/1/4 unit 0 family inet address 192.0.2.1/24
set interfaces ge-1/1/4 unit 0 family iso
set interfaces ge-1/1/4 unit 0 family mpls
set interfaces ge-1/1/6 gigether-options 802.3ad ae12
set interfaces ge-1/2/1 flexible-vlan-tagging
set interfaces ge-1/2/1 encapsulation flexible-ethernet-services
set interfaces ge-1/2/1 esi 00:33:33:33:33:33:33:33:33:33
set interfaces ge-1/2/1 esi all-active
set interfaces ge-1/2/1 unit 0 encapsulation vlan-bridge
set interfaces ge-1/2/1 unit 0 vlan-id 30
set interfaces ge-1/2/1 unit 130 family bridge interface-mode trunk
set interfaces ge-1/2/1 unit 130 family bridge vlan-id-list 130
set interfaces ge-1/2/1 unit 230 family bridge interface-mode trunk
set interfaces ge-1/2/1 unit 230 family bridge vlan-id-list 230
set interfaces ge-1/2/2 flexible-vlan-tagging
set interfaces ge-1/2/2 encapsulation flexible-ethernet-services
set interfaces ge-1/2/2 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ge-1/2/2 esi all-active
set interfaces ge-1/2/2 unit 0 encapsulation vlan-bridge
set interfaces ge-1/2/2 unit 0 vlan-id 20
set interfaces ge-1/2/2 unit 120 family bridge interface-mode trunk
set interfaces ge-1/2/2 unit 120 family bridge vlan-id-list 120
set interfaces ge-1/2/2 unit 220 family bridge interface-mode trunk
set interfaces ge-1/2/2 unit 220 family bridge vlan-id-list 220
set interfaces ge-1/2/3 gigether-options 802.3ad ae0
set interfaces ge-1/2/4 gigether-options 802.3ad ae0
set interfaces ge-1/3/0 flexible-vlan-tagging
set interfaces ge-1/3/0 encapsulation flexible-ethernet-services
set interfaces ge-1/3/0 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/0 unit 0 vlan-id 10
set interfaces ge-1/3/0 unit 100 family bridge interface-mode trunk
set interfaces ge-1/3/0 unit 100 family bridge vlan-id-list 110
set interfaces ge-1/3/0 unit 100 family bridge vlan-id-list 120
set interfaces ge-1/3/0 unit 100 family bridge vlan-id-list 130
set interfaces ge-1/3/0 unit 200 family bridge interface-mode trunk
set interfaces ge-1/3/0 unit 200 family bridge vlan-id-list 210
set interfaces ge-1/3/0 unit 200 family bridge vlan-id-list 220
set interfaces ge-1/3/0 unit 200 family bridge vlan-id-list 230
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 family bridge interface-mode trunk
set interfaces ae0 unit 110 family bridge vlan-id-list 110
set interfaces ae0 unit 210 family bridge interface-mode trunk
set interfaces ae0 unit 210 family bridge vlan-id-list 210
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
```

```
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
set interfaces ae12 unit 0 family inet address 198.51.100.12/24
set interfaces ae12 unit 0 family iso
set interfaces ae12 unit 0 family mpls
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 198.51.100.13/24
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces irb unit 0 family inet address 192.0.2.9/24
set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99
set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.1/24 primary
set interfaces lo0 unit 0 family iso
set routing-options router-id 198.51.100.1
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe1tope2 from 198.51.100.1
set protocols mpls label-switched-path pe1tope2 to 198.51.100.2
set protocols mpls label-switched-path pe1tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe1tope3 from 198.51.100.1
set protocols mpls label-switched-path pe1tope3 to 198.51.100.3
set protocols mpls label-switched-path pe1tope3 primary direct_to_pe3
set protocols mpls label-switched-path pe1tope4 from 198.51.100.1
set protocols mpls label-switched-path pe1tope4 to 198.51.100.4
set protocols mpls label-switched-path pe1tope4 primary direct_to_pe4
set protocols mpls path pe4_to_pe3 198.51.100.4 strict
set protocols mpls path pe4_to_pe3 198.51.100.3 strict
set protocols mpls path direct_to_pe2 198.51.100.5 strict
set protocols mpls path direct_to_pe3 198.51.100.6 strict
set protocols mpls path direct_to_pe4 198.51.100.9 strict
set protocols mpls path pe2_to_pe3 198.51.100.2 strict
set protocols mpls path pe2_to_pe3 198.51.100.3 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal
set protocols bgp group RR local-address 198.51.100.1
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
```

```
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols evpn
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-1/2/1.130
set routing-instances VS-1 interface ge-1/2/2.120
set routing-instances VS-1 interface ge-1/3/0.100
set routing-instances VS-1 interface ae0.110
set routing-instances VS-1 route-distinguisher 198.51.100.1:101
set routing-instances VS-1 vrf-target target:100:101
set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise
set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-1/2/1.230
set routing-instances VS-2 interface ge-1/2/2.220
set routing-instances VS-2 interface ge-1/3/0.200
set routing-instances VS-2 interface ae0.210
set routing-instances VS-2 route-distinguisher 198.51.100.1:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-1/2/1.0
set routing-instances mhevpn interface ge-1/2/2.0
set routing-instances mhevpn interface ge-1/3/0.0
set routing-instances mhevpn interface ae0.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.1:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.1:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label
```

PE2

```
set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-0/0/5 gigether-options 802.3ad ae12
set interfaces ge-0/1/0 gigether-options 802.3ad ae12
set interfaces ge-0/1/1 flexible-vlan-tagging
set interfaces ge-0/1/1 encapsulation flexible-ethernet-services
set interfaces ge-0/1/1 esi 00:33:33:33:33:33:33:33:33:33
set interfaces ge-0/1/1 esi all-active
set interfaces ge-0/1/1 unit 0 encapsulation vlan-bridge
set interfaces ge-0/1/1 unit 0 vlan-id 30
set interfaces ge-0/1/1 unit 130 family bridge interface-mode trunk
set interfaces ge-0/1/1 unit 130 family bridge vlan-id-list 130
set interfaces ge-0/1/1 unit 230 family bridge interface-mode trunk
set interfaces ge-0/1/1 unit 230 family bridge vlan-id-list 230
set interfaces ge-0/1/2 flexible-vlan-tagging
set interfaces ge-0/1/2 encapsulation flexible-ethernet-services
set interfaces ge-0/1/2 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ge-0/1/2 esi all-active
set interfaces ge-0/1/2 unit 0 encapsulation vlan-bridge
set interfaces ge-0/1/2 unit 0 vlan-id 20
set interfaces ge-0/1/2 unit 120 family bridge interface-mode trunk
set interfaces ge-0/1/2 unit 120 family bridge vlan-id-list 120
set interfaces ge-0/1/2 unit 220 family bridge interface-mode trunk
set interfaces ge-0/1/2 unit 220 family bridge vlan-id-list 220
set interfaces ge-0/1/3 unit 0 family inet address 192.0.2.3/24
set interfaces ge-0/1/3 unit 0 family iso
set interfaces ge-0/1/3 unit 0 family mpls
set interfaces ge-0/1/4 gigether-options 802.3ad ae0
set interfaces ge-0/1/6 flexible-vlan-tagging
set interfaces ge-0/1/6 encapsulation flexible-ethernet-services
set interfaces ge-0/1/6 unit 0 encapsulation vlan-bridge
set interfaces ge-0/1/6 unit 0 vlan-id 10
set interfaces ge-0/1/6 unit 100 family bridge interface-mode trunk
set interfaces ge-0/1/6 unit 100 family bridge vlan-id-list 110
set interfaces ge-0/1/6 unit 100 family bridge vlan-id-list 120
set interfaces ge-0/1/6 unit 100 family bridge vlan-id-list 130
set interfaces ge-0/1/6 unit 200 family bridge interface-mode trunk
set interfaces ge-0/1/6 unit 200 family bridge vlan-id-list 210
set interfaces ge-0/1/6 unit 200 family bridge vlan-id-list 220
set interfaces ge-0/1/6 unit 200 family bridge vlan-id-list 230
set interfaces ge-0/1/9 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 family bridge interface-mode trunk
set interfaces ae0 unit 110 family bridge vlan-id-list 110
set interfaces ae0 unit 210 family bridge interface-mode trunk
set interfaces ae0 unit 210 family bridge vlan-id-list 210
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
```

```
set interfaces ae12 unit 0 family inet address 198.51.100.5/24
set interfaces ae12 unit 0 family iso
set interfaces ae12 unit 0 family mpls
set interfaces irb unit 0 family inet address 192.0.2.9/24
set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99
set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.2/32 primary
set interfaces lo0 unit 0 family iso
set routing-options router-id 198.51.100.2
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe2tope1 from 198.51.100.2
set protocols mpls label-switched-path pe2tope1 to 198.51.100.1
set protocols mpls label-switched-path pe2tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe2tope3 from 198.51.100.2
set protocols mpls label-switched-path pe2tope3 to 198.51.100.3
set protocols mpls label-switched-path pe2tope3 primary direct_to_pe3
set protocols mpls label-switched-path pe2tope4 from 198.51.100.2
set protocols mpls label-switched-path pe2tope4 to 198.51.100.4
set protocols mpls label-switched-path pe2tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 198.51.100.12 strict
set protocols mpls path direct_to_pe3 198.51.100.7 strict
set protocols mpls path direct_to_pe4 198.51.100.8 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal
set protocols bgp group RR local-address 198.51.100.2
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-0/1/1.130
set routing-instances VS-1 interface ge-0/1/2.120
set routing-instances VS-1 interface ge-0/1/6.100
set routing-instances VS-1 interface ae0.110
set routing-instances VS-1 route-distinguisher 198.51.100.2:101
set routing-instances VS-1 vrf-target target:100:101
set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
```

```
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise
set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-0/1/1.230
set routing-instances VS-2 interface ge-0/1/2.220
set routing-instances VS-2 interface ge-0/1/6.200
set routing-instances VS-2 interface ae0.210
set routing-instances VS-2 route-distinguisher 198.51.100.2:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-0/1/1.0
set routing-instances mhevpn interface ge-0/1/2.0
set routing-instances mhevpn interface ge-0/1/6.0
set routing-instances mhevpn interface ae0.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.2:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.2:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label
```

PE3

```
set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-2/0/1 gigether-options 802.3ad ae13
set interfaces ge-2/0/2 gigether-options 802.3ad ae13
set interfaces ge-2/0/4 gigether-options 802.3ad ae0
set interfaces ge-2/0/9 flexible-vlan-tagging
set interfaces ge-2/0/9 encapsulation flexible-ethernet-services
set interfaces ge-2/0/9 unit 0 encapsulation vlan-bridge
set interfaces ge-2/0/9 unit 0 vlan-id 10
set interfaces ge-2/0/9 unit 100 family bridge interface-mode trunk
set interfaces ge-2/0/9 unit 100 family bridge vlan-id-list 110
set interfaces ge-2/0/9 unit 100 family bridge vlan-id-list 120
set interfaces ge-2/0/9 unit 100 family bridge vlan-id-list 130
set interfaces ge-2/0/9 unit 200 family bridge interface-mode trunk
```

```
set interfaces ge-2/0/9 unit 200 family bridge vlan-id-list 210
set interfaces ge-2/0/9 unit 200 family bridge vlan-id-list 220
set interfaces ge-2/0/9 unit 200 family bridge vlan-id-list 230
set interfaces ge-2/1/0 unit 0 family inet address 192.0.2.5/24
set interfaces ge-2/1/0 unit 0 family iso
set interfaces ge-2/1/0 unit 0 family mpls
set interfaces ge-2/1/6 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 110 family bridge interface-mode trunk
set interfaces ae0 unit 110 family bridge vlan-id-list 110
set interfaces ae0 unit 210 family bridge interface-mode trunk
set interfaces ae0 unit 210 family bridge vlan-id-list 210
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 198.51.100.6/24
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces irb unit 0 family inet address 192.0.2.9/24
set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99
set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.3/32 primary
set interfaces lo0 unit 0 family iso
set routing-options router-id 198.51.100.3
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe3tope1 from 198.51.100.3
set protocols mpls label-switched-path pe3tope1 to 198.51.100.1
set protocols mpls label-switched-path pe3tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe3tope2 from 198.51.100.3
set protocols mpls label-switched-path pe3tope2 to 198.51.100.2
set protocols mpls label-switched-path pe3tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe3tope4 from 198.51.100.3
set protocols mpls label-switched-path pe3tope4 to 198.51.100.4
set protocols mpls label-switched-path pe3tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 198.51.100.13 strict
set protocols mpls path direct_to_pe2 198.51.100.10 strict
set protocols mpls path direct_to_pe4 198.51.100.11 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal
```

```
set protocols bgp group RR local-address 198.51.100.3
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-2/0/9.100
set routing-instances VS-1 interface ae0.110
set routing-instances VS-1 route-distinguisher 198.51.100.3:101
set routing-instances VS-1 vrf-target target:100:101
set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise
set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-2/0/9.200
set routing-instances VS-2 interface ae0.210
set routing-instances VS-2 route-distinguisher 198.51.100.3:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-2/0/9.0
set routing-instances mhevpn interface ae0.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.3:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.3:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label
```

PE4

```
set chassis aggregated-devices ethernet device-count 20
set chassis network-services enhanced-ip
set interfaces ge-1/0/3 unit 0 family inet address 192.0.2.7/24
set interfaces ge-1/0/3 unit 0 family iso
set interfaces ge-1/0/3 unit 0 family mpls
set interfaces ge-1/1/1 flexible-vlan-tagging
set interfaces ge-1/1/1 encapsulation flexible-ethernet-services
set interfaces ge-1/1/1 esi 00:44:44:44:44:44:44:44:44:44
set interfaces ge-1/1/1 esi all-active
set interfaces ge-1/1/1 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/1 unit 0 vlan-id 10
set interfaces ge-1/1/1 unit 100 family bridge interface-mode trunk
set interfaces ge-1/1/1 unit 100 family bridge vlan-id-list 110
set interfaces ge-1/1/1 unit 100 family bridge vlan-id-list 120
set interfaces ge-1/1/1 unit 100 family bridge vlan-id-list 130
set interfaces ge-1/1/1 unit 200 family bridge interface-mode trunk
set interfaces ge-1/1/1 unit 200 family bridge vlan-id-list 210
set interfaces ge-1/1/1 unit 200 family bridge vlan-id-list 220
set interfaces ge-1/1/1 unit 200 family bridge vlan-id-list 230
set interfaces ge-1/1/9 flexible-vlan-tagging
set interfaces ge-1/1/9 encapsulation flexible-ethernet-services
set interfaces ge-1/1/9 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/9 unit 0 vlan-id 10
set interfaces ge-1/1/9 unit 100 family bridge interface-mode trunk
set interfaces ge-1/1/9 unit 100 family bridge vlan-id-list 110
set interfaces ge-1/1/9 unit 100 family bridge vlan-id-list 120
set interfaces ge-1/1/9 unit 100 family bridge vlan-id-list 130
set interfaces ge-1/1/9 unit 200 family bridge interface-mode trunk
set interfaces ge-1/1/9 unit 200 family bridge vlan-id-list 210
set interfaces ge-1/1/9 unit 200 family bridge vlan-id-list 220
set interfaces ge-1/1/9 unit 200 family bridge vlan-id-list 230
set interfaces irb unit 0 family inet address 192.0.2.9/24
set interfaces irb unit 0 mac 00:99:99:99:01:99
set interfaces irb unit 1 family inet address 192.0.2.10/24
set interfaces irb unit 1 mac 00:99:99:99:02:99
set interfaces irb unit 2 family inet address 192.0.2.11/24
set interfaces irb unit 2 mac 00:99:99:99:03:99
set interfaces irb unit 10 family inet address 192.0.2.12/24
set interfaces irb unit 10 mac 00:99:99:99:01:90
set interfaces lo0 unit 0 family inet address 198.51.100.4/32 primary
set routing-options router-id 198.51.100.4
set routing-options autonomous-system 65221
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe4tope1 from 198.51.100.4
set protocols mpls label-switched-path pe4tope1 to 198.51.100.1
set protocols mpls label-switched-path pe4tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe4tope2 from 198.51.100.4
set protocols mpls label-switched-path pe4tope2 to 198.51.100.2
set protocols mpls label-switched-path pe4tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe4tope3 from 198.51.100.4
set protocols mpls label-switched-path pe4tope3 to 198.51.100.3
set protocols mpls label-switched-path pe4tope3 primary direct_to_pe3
set protocols mpls path pe2_to_pe3 198.51.100.2 strict
```

```
set protocols mpls path pe2_to_pe3 198.51.100.3 strict
set protocols mpls path direct_to_pe1 198.51.100.14 strict
set protocols mpls path direct_to_pe2 198.51.100.15 strict
set protocols mpls path direct_to_pe3 198.51.100.16 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR type internal
set protocols bgp group RR local-address 198.51.100.4
set protocols bgp group RR family evpn signaling
set protocols bgp group RR neighbor 203.0.113.0
set protocols isis level 1 disable
set protocols isis interface all level 2 metric 10
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols l2-learning global-mac-table-aging-time 18000
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-instances VS-1 instance-type virtual-switch
set routing-instances VS-1 interface ge-1/1/1.100
set routing-instances VS-1 interface ge-1/1/9.100
set routing-instances VS-1 route-distinguisher 198.51.100.4:101
set routing-instances VS-1 vrf-target target:100:101
set routing-instances VS-1 protocols evpn extended-vlan-list 110
set routing-instances VS-1 protocols evpn extended-vlan-list 120
set routing-instances VS-1 protocols evpn extended-vlan-list 130
set routing-instances VS-1 protocols evpn default-gateway do-not-advertise
set routing-instances VS-1 bridge-domains bd-110 vlan-id 110
set routing-instances VS-1 bridge-domains bd-110 routing-interface irb.0
set routing-instances VS-1 bridge-domains bd-120 vlan-id 120
set routing-instances VS-1 bridge-domains bd-120 routing-interface irb.1
set routing-instances VS-1 bridge-domains bd-130 vlan-id 130
set routing-instances VS-1 bridge-domains bd-130 routing-interface irb.2
set routing-instances VS-2 instance-type virtual-switch
set routing-instances VS-2 interface ge-1/1/1.200
set routing-instances VS-2 interface ge-1/1/9.200
set routing-instances VS-2 route-distinguisher 198.51.100.4:201
set routing-instances VS-2 vrf-target target:100:201
set routing-instances VS-2 protocols evpn extended-vlan-list 210
set routing-instances VS-2 protocols evpn extended-vlan-list 220
set routing-instances VS-2 protocols evpn extended-vlan-list 230
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
set routing-instances mhevpn instance-type evpn
set routing-instances mhevpn vlan-id 10
set routing-instances mhevpn interface ge-1/1/1.0
set routing-instances mhevpn interface ge-1/1/9.0
set routing-instances mhevpn routing-interface irb.10
set routing-instances mhevpn route-distinguisher 198.51.100.4:1
set routing-instances mhevpn vrf-target target:100:1
set routing-instances mhevpn protocols evpn default-gateway do-not-advertise
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
```

```
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
set routing-instances vrf interface irb.10
set routing-instances vrf route-distinguisher 198.51.100.4:11
set routing-instances vrf vrf-target target:100:11
set routing-instances vrf vrf-table-label
```

P (RR)

```
set interfaces ge-1/0/5 unit 0 family inet address 192.0.2.8/24
set interfaces ge-1/1/2 unit 0 family inet address 192.0.2.6/24
set interfaces ge-1/1/3 unit 0 family inet address 192.0.2.4/24
set interfaces ge-1/1/4 unit 0 family inet address 192.0.2.2/24
set interfaces lo0 unit 0 family inet address 203.0.113.0
set protocols bgp group RR type internal
set protocols bgp group RR local-address 203.0.113.0
set protocols bgp group RR family evpn signaling
set protocols bgp group RR cluster 1.2.3.4
set protocols bgp group RR neighbor 198.51.100.1
set protocols bgp group RR neighbor 198.51.100.2
set protocols bgp group RR neighbor 198.51.100.3
set protocols bgp group RR neighbor 198.51.100.4
set protocols isis interface all level 1 disable
set protocols ldp interface all
set routing-options router-id 203.0.113.0
set routing-options autonomous-system 65221
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for all other multihomed PE routers after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the MX Series router to operate in the enhanced-ip mode because the EVPN active-active functionality is supported on routers with MPCs and MIC interfaces only.

   A system reboot is required on committing this configuration.

   ```
   [edit chassis]
   user@PE1# set network-services enhanced-ip
   ```

2. Specify the number of aggregated Ethernet interfaces to be created.

   ```
   [edit chassis]
   user@PE1# set aggregated-devices ethernet device-count 20
   ```

3. Configure Router PE1 interfaces within the ae0 aggregated bundle toward the multihomed customer site, Router CE10.

   a. Assign interfaces ge-1/2/3 and ge-1/2/4 within the ae0 aggregated bundle.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/3 gigether-options 802.3ad ae0
   user@PE1# set ge-1/2/4 gigether-options 802.3ad ae0
   ```

   b. Configure the ae0 aggregated bundle parameters for VLAN tagging and encapsulation.

   ```
   [edit interfaces]
   user@PE1# set ae0 flexible-vlan-tagging
   user@PE1# set ae0 encapsulation flexible-ethernet-services
   ```

   c. Assign an ESI value for the first Ethernet segment and enable EVPN active-active multihoming for the ae0 aggregated bundle.

   ```
   [edit interfaces]
   user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11:11
   user@PE1# set ae0 esi all-active
   ```

   d. Configure a trunk interface on the bridge network for the ae0 aggregated bundle.

   ```
   [edit interfaces]
   user@PE1# set ae0 unit 0 encapsulation vlan-bridge
   user@PE1# set ae0 unit 0 vlan-id 10
   user@PE1# set ae0 unit 110 family bridge interface-mode trunk
   user@PE1# set ae0 unit 110 family bridge vlan-id-list 110
   user@PE1# set ae0 unit 210 family bridge interface-mode trunk
   user@PE1# set ae0 unit 210 family bridge vlan-id-list 210
   ```

4. Configure the other Router PE1 interfaces toward the multihomed customer site, Router CE10.

   a. Configure the VLAN tagging and encapsulation parameters for the ge-1/2/2 PE1 interface.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/2 flexible-vlan-tagging
   user@PE1# set ge-1/2/2 encapsulation flexible-ethernet-services
   ```

   b. Assign an ESI value for the second Ethernet segment and enable EVPN active-active multihoming for the ge-1/2/2 PE1 interface.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/2 esi 00:22:22:22:22:22:22:22:22:22
   user@PE1# set ge-1/2/2 esi all-active
   ```

   c. Configure a trunk interface on the bridge network for the ge-1/2/2 PE1 interface.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/2 unit 0 encapsulation vlan-bridge
   ```

```
user@PE1# set ge-1/2/2 unit 0 vlan-id 20
user@PE1# set ge-1/2/2 unit 120 family bridge interface-mode trunk
user@PE1#  set ge-1/2/2 unit 120 family bridge vlan-id-list 120
user@PE1# set ge-1/2/2 unit 220 family bridge interface-mode trunk
user@PE1# set ge-1/2/2 unit 220 family bridge vlan-id-list 220
```

d. Configure the VLAN tagging and encapsulation parameters for the ge-1/2/1 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/1 flexible-vlan-tagging
user@PE1# set ge-1/2/1 encapsulation flexible-ethernet-services
```

e. Assign an ESI value for the third Ethernet segment and enable EVPN active-active multihoming for the ge-1/2/1 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/1 esi 00:33:33:33:33:33:33:33:33:33
user@PE1# set ge-1/2/1 esi all-active
```

f. Configure a trunk interface on the bridge network for the ge-1/2/1 PE1 interface.

```
[edit interfaces]
user@PE1# set ge-1/2/1 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/2/1 unit 0 vlan-id 30
user@PE1# set ge-1/2/1 unit 130 family bridge interface-mode trunk
user@PE1# set ge-1/2/1 unit 130 family bridge vlan-id-list 130
user@PE1# set ge-1/2/1 unit 230 family bridge interface-mode trunk
user@PE1# set ge-1/2/1 unit 230 family bridge vlan-id-list 230
```

5. Configure Router PE1 interfaces toward Router PE2.

a. Assign the interfaces ge-1/0/5 and ge-1/1/6 within the ae12 aggregated bundle.

```
[edit interfaces]
user@PE1# set ge-1/0/5 gigether-options 802.3ad ae12
user@PE1# set ge-1/1/6 gigether-options 802.3ad ae12
```

b. Specify the minimum number of links for the ae12 aggregated bundle to be labeled “up”.

```
[edit interfaces]
user@PE1# set ae12 aggregated-ether-options minimum-links 1
```

c. Assign an IP address for the ae12 aggregated bundle and enable MPLS and IS-IS protocol families on the bundle.

```
[edit interfaces]
user@PE1# set ae12 unit 0 family inet address 198.51.100.12/24
user@PE1# set ae12 unit 0 family iso
user@PE1# set ae12 unit 0 family mpls
```

d. Configure the VLAN tagging and encapsulation parameters for the ae12 aggregated bundle and assign VLAN ID 1200 for the bundle.

```
[edit interfaces]
user@PE1# set ae12 flexible-vlan-tagging
user@PE1# set ae12 encapsulation flexible-ethernet-services
user@PE1# set ae12 unit 0 vlan-id 1200
```

6. Configure Router PE1 interfaces toward Router PE3.

   a. Assign the interfaces ge-1/0/3 and ge-1/0/4 within the ae13 aggregated bundle.

   ```
   [edit interfaces]
   user@PE1# set ge-1/0/3 gigether-options 802.3ad ae13
   user@PE1# set ge-1/0/4 gigether-options 802.3ad ae13
   ```

   b. Specify the minimum number of links for the ae13 aggregated bundle to be labeled “up”.

   ```
   [edit interfaces]
   user@PE1# set ae13 aggregated-ether-options minimum-links 1
   ```

   c. Assign an IP address for the ae13 aggregated bundle and enable MPLS and IS-IS protocol families on the bundle.

   ```
   [edit interfaces]
   user@PE1# set ae13 unit 0 family inet address 198.51.100.13/24
   user@PE1# set ae13 unit 0 family iso
   user@PE1# set ae13 unit 0 family mpls
   ```

   d. Configure the VLAN tagging and encapsulation parameters for the ae12 aggregated bundle and assign the inner and outer VLAN tags for the bundle.

   ```
   [edit interfaces]
   user@PE1# set ae13 flexible-vlan-tagging
   user@PE1# set ae13 encapsulation flexible-ethernet-services
   user@PE1# set ae13 unit 0 vlan-tags outer 1300
   user@PE1# set ae13 unit 0 vlan-tags inner 13
   ```

7. Configure the Router PE1 interface toward the single-homed customer site, Router CE1.

   a. Configure the VLAN tagging and encapsulation parameters for the ge-1/3/0 PE1 interface.

   ```
   [edit interfaces]
   user@PE1# set ge-1/3/0 flexible-vlan-tagging
   user@PE1# set ge-1/3/0 encapsulation flexible-ethernet-services
   ```

   b. Configure a trunk interface on the bridge network for the ge-1/3/0 PE1 interface.

   ```
   [edit interfaces]
   user@PE1# set ge-1/3/0 unit 0 encapsulation vlan-bridge
   user@PE1# set ge-1/3/0 unit 0 vlan-id 10
   user@PE1# set ge-1/3/0 unit 100 family bridge interface-mode trunk
   user@PE1# set ge-1/3/0 unit 100 family bridge vlan-id-list 110
   user@PE1# set ge-1/3/0 unit 100 family bridge vlan-id-list 120
   ```

```
user@PE1# set ge-1/3/0 unit 100 family bridge vlan-id-list 130
user@PE1# set ge-1/3/0 unit 200 family bridge interface-mode trunk
user@PE1# set ge-1/3/0 unit 200 family bridge vlan-id-list 210
user@PE1# set ge-1/3/0 unit 200 family bridge vlan-id-list 220
user@PE1# set ge-1/3/0 unit 200 family bridge vlan-id-list 230
```

8. Configure the Router PE1 interface toward Router P (RR) and enable the MPLS and IS-IS protocol families for the interface.

```
[edit interfaces]
user@PE1# set ge-1/1/4 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-1/1/4 unit 0 family iso
user@PE1# set ge-1/1/4 unit 0 family mpls
```

9. Configure an IRB interface for Router PE1.

```
[edit interfaces]
user@PE1# set irb unit 0 family inet address 192.0.2.9/24
user@PE1# set irb unit 0 mac 00:99:99:99:01:99
user@PE1# set irb unit 1 family inet address 192.0.2.10/24
user@PE1# set irb unit 1 mac 00:99:99:99:02:99
user@PE1# set irb unit 2 family inet address 192.0.2.11/24
user@PE1# set irb unit 2 mac 00:99:99:99:03:99
user@PE1# set irb unit 10 family inet address 192.0.2.12/24
user@PE1# set irb unit 10 mac 00:99:99:99:01:90
```

10. Configure the loopback interface for Router PE1.

```
[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 198.51.100.1/24 primary
user@PE1# set lo0 unit 0 family iso
```

11. Assign a router ID and the autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set router-id 198.51.100.1
user@PE1# set autonomous-system 65221
```

12. Assign a load-balancing policy to the forwarding table of Router PE1.

```
[edit routing-options]
user@PE1# set forwarding-table export load-balancing-policy
```

13. Configure IS-IS on Router PE1.

```
[edit protocols]
user@PE1# set isis level 1 disable
user@PE1# set isis interface all level 2 metric 10
user@PE1# set isis interface fxp0.0 disable
```

```
user@PE1# set isis interface lo0.0 level 2 metric 0
```

14. Configure an internal BGP group for Router PE1 to peer with route reflector, Router P.

```
[edit protocols]
user@PE1# set bgp group RR type internal
user@PE1# set bgp group RR local-address 198.51.100.1
user@PE1# set bgp group RR neighbor 203.0.113.0
```

15. Enable EVPN signaling for the RR BGP group on Router PE1.

```
[edit protocols]
user@PE1# set bgp group RR family evpn signaling
```

16. Configure RSVP, LDP, MPLS, EVPN, and L2 learning on Router PE1.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
user@PE1# set ldp deaggregate
user@PE1# set ldp interface all
user@PE1# set ldp interface fxp0.0 disable
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable
user@PE1# set evpn
user@PE1# set l2-learning global-mac-table-aging-time 18000
```

17. Configure label-switched paths between the PE routers.

```
[edit protocols]
user@PE1# set mpls label-switched-path pe1tope2 from 198.51.100.1
user@PE1# set mpls label-switched-path pe1tope2 to 198.51.100.2
user@PE1# set mpls label-switched-path pe1tope2 primary direct_to_pe2
user@PE1# set mpls label-switched-path pe1tope3 from 198.51.100.1
user@PE1# set mpls label-switched-path pe1tope3 to 198.51.100.3
user@PE1# set mpls label-switched-path pe1tope3 primary direct_to_pe3
user@PE1# set mpls label-switched-path pe1tope4 from 198.51.100.1
user@PE1# set mpls label-switched-path pe1tope4 to 198.51.100.4
user@PE1# set mpls label-switched-path pe1tope4 primary direct_to_pe4
```

18. Configure MPLS paths from Router PE1 to other PE routers.

```
user@PE1# set mpls path pe4_to_pe3 198.51.100.4 strict
user@PE1# set mpls path pe4_to_pe3 198.51.100.3 strict
user@PE1# set mpls path direct_to_pe2 198.51.100.5 strict
user@PE1# set mpls path direct_to_pe3 198.51.100.6 strict
user@PE1# set mpls path direct_to_pe4 198.51.100.9 strict
user@PE1# set mpls path pe2_to_pe3 198.51.100.2 strict
```

```
user@PE1# set mpls path pe2_to_pe3 198.51.100.3 strict
```

19. Configure the load-balancing policy to enable load balancing per packet.

```
[edit policy-options]
user@PE1# set policy-statement load-balancing-policy then load-balance per-packet
```

20. Configure the first virtual switch routing instance.

    a. Configure the routing-instance type and assign Router PE1 interfaces to the routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 instance-type virtual-switch
user@PE1# set VS-1 interface ge-1/2/1.130
user@PE1# set VS-1 interface ge-1/2/2.120
user@PE1# set VS-1 interface ge-1/3/0.100
user@PE1# set VS-1 interface ae0.110
```

    b. Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the VS-1 routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 route-distinguisher 198.51.100.1:101
user@PE1# set VS-1 vrf-target target:100:101
```

    c. Configure EVPN and assign VLANs to the VS-1 routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 protocols evpn extended-vlan-list 110
user@PE1# set VS-1 protocols evpn extended-vlan-list 120
user@PE1# set VS-1 protocols evpn extended-vlan-list 130
user@PE1# set VS-1 protocols evpn default-gateway do-not-advertise
```

    d. Configure the bridge domains and their associated VLANs and IRB interfaces for the VS-1 routing instance.

```
[edit routing-instances]
user@PE1# set VS-1 bridge-domains bd-110 vlan-id 110
user@PE1# set VS-1 bridge-domains bd-110 routing-interface irb.0
user@PE1# set VS-1 bridge-domains bd-120 vlan-id 120
user@PE1# set VS-1 bridge-domains bd-120 routing-interface irb.1
user@PE1# set VS-1 bridge-domains bd-130 vlan-id 130
user@PE1# set VS-1 bridge-domains bd-130 routing-interface irb.2
```

21. Configure the second virtual switch routing instance.

    a. Configure the routing-instance type and assign Router PE1 interfaces to the routing instance.

```
[edit routing-instances]
user@PE1# set VS-2 instance-type virtual-switch
```

```
user@PE1# set VS-2 interface ge-1/2/1.230
user@PE1# set VS-2 interface ge-1/2/2.220
user@PE1# set VS-2 interface ge-1/3/0.200
user@PE1# set VS-2 interface ae0.210
```

b. Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the VS-2 routing instance.

```
[edit routing-instances]
user@PE1# set VS-2 route-distinguisher 198.51.100.1:201
user@PE1# set VS-2 vrf-target target:100:201
```

c. Configure EVPN and assign VLANs to the VS-2 routing instance.

```
[edit routing-instances]
user@PE1# set VS-2 protocols evpn extended-vlan-list 210
user@PE1# set VS-2 protocols evpn extended-vlan-list 220
user@PE1# set VS-2 protocols evpn extended-vlan-list 230
```

d. Configure the bridge domains and their associated VLANs for the VS-2 routing instance.

```
[edit routing-instances]
user@PE1# set routing-instances VS-2 bridge-domains bd-a vlan-id-list 210
user@PE1# set routing-instances VS-2 bridge-domains bd-a vlan-id-list 220
user@PE1# set routing-instances VS-2 bridge-domains bd-a vlan-id-list 230
```

22. Configure the multihomed EVPN routing instance.

a. Configure the routing-instance type and assign VLANs and Router PE1 interfaces to the routing instance.

```
[edit routing-instances]
user@PE1# set mhevpn instance-type evpn
user@PE1# set mhevpn vlan-id 10
user@PE1# set mhevpn interface ge-1/2/1.0
user@PE1# set mhevpn interface ge-1/2/2.0
user@PE1# set mhevpn interface ge-1/3/0.0
user@PE1# set mhevpn interface ae0.0
user@PE1# set mhevpn routing-interface irb.10
```

b. Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the mhevpn routing instance.

```
[edit routing-instances]
user@PE1# set mhevpn route-distinguisher 198.51.100.1:1
user@PE1# set mhevpn vrf-target target:100:1
```

c. Configure EVPN to the mhevpn routing instance.

```
[edit routing-instances]
```

```
user@PE1# set routing-instances mhevpn protocols evpn default-gateway
    do-not-advertise
```

23. Configure the default routing instance.

    a. Configure the routing-instance type and assign IRB interfaces to the routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface irb.0
user@PE1# set vrf interface irb.1
user@PE1# set vrf interface irb.2
user@PE1# set vrf interface irb.10
```

    b. Configure the route distinguisher and the VPN routing and forwarding (VRF) target for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 198.51.100.1:11
user@PE1# set vrf vrf-target target:100:11
user@PE1# set vrf vrf-table-label
```

Configuration on EX9200 Switches

Step-by-Step Procedure

Several configuration statements used to configure active-active mode differ on EX9200 switches from those used on MX Series routers. This procedure shows which configuration statements are specific to EX9200 switches. All other configuration in this example applies both to EX9200 switches and MX Series routers.

1. To configure a trunk interface, include the **family ethernet-switching** statements instead of the **family bridge** statements in all occurrences.

```
[edit interfaces]
user@PE#1# set interfaces ge-1/2/1 unit 130 family ethernet-switching
    interface-mode trunk
```

2. To configure the Layer 2 Ethernet switching domain, include the **vlan members** (*vlan-id* | *name*) statement instead of the **vlan-id-list** *vlan-id* statement in all occurrences.

```
[edit interfaces]
user@PE#1# set interfaces ge-1/2/1 unit 130 family ethernet-switching vlan members
    130
```

3. To configure the VLAN domain and associated VLANs for each routing instance, include the **vlans** *name* statement, instead of the **bridge-domains** statement in all occurrences.

```
[edit]
```

```
user@PE#1# set routing-instances VS-1 vlans bd-110 vlan-id 110
```

4. To configure the IRB interface in each routing instance, include the **l3-interface** ***irb-interface-name*** statement instead of the **routing-interface** statement in all occurrences.

```
[edit]
user@PE#1# set routing-instances VS-1 vlans bd-110 l3-interface irb.0
```

Results

From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1 show chassis
aggregated-devices {
   ethernet {
      device-count 20;
   }
}
network-services enhanced-ip;
```

```
user@PE1 show interfaces
ge-1/0/3 {
   gigether-options {
      802.3ad ae13;
   }
}
ge-1/0/4 {
   gigether-options {
      802.3ad ae13;
   }
}
ge-1/0/5 {
   gigether-options {
      802.3ad ae12;
   }
}
ge-1/1/4 {
   unit 0 {
      family inet {
         address 192.0.2.1/24;
      }
      family iso;
      family mpls;
   }
}
ge-1/1/6 {
```

```
    gigether-options {
      802.3ad ae12;
    }
}
ge-1/2/1 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:33:33:33:33:33:33:33:33:33;
      all-active;
    }
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id 30;
    }
    unit 130 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 130;
      }
    }
    unit 230 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 230;
      }
    }
}
ge-1/2/2 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:22:22:22:22:22:22:22:22:22;
      all-active;
    }
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id 20;
    }
    unit 120 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 120;
      }
    }
    unit 220 {
      family bridge {
        interface-mode trunk;
        vlan-id-list 220;
      }
    }
}
ge-1/2/3 {
    gigether-options {
      802.3ad ae0;
```

```
    }
}
ge-1/2/4 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/3/0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
    unit 100 {
        family bridge {
            interface-mode trunk;
            vlan-id-list [ 110 120 130 ];
        }
    }
    unit 200 {
        family bridge {
            interface-mode trunk;
            vlan-id-list [ 210 220 230 ];
        }
    }
}
ae0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
        00:11:11:11:11:11:11:11:11:11;
        all-active;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
    unit 110 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 110;
        }
    }
    unit 210 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 210;
        }
    }
}
ae12 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    aggregated-ether-options {
```

```
      minimum-links 1;
    }
    unit 0 {
      vlan-id 1200;
      family inet {
        address 198.51.100.12/24;
      }
      family iso;
      family mpls;
    }
}
ae13 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    aggregated-ether-options {
      minimum-links 1;
    }
    unit 0 {
      vlan-tags outer 1300 inner 13;
      family inet {
        address 198.51.100.13/24;
      }
      family iso;
      family mpls;
    }
}
irb {
    unit 0 {
      family inet {
        address 192.0.2.9/24;
      }
      mac 00:99:99:99:01:99;
    }
    unit 1 {
      family inet {
        address 192.0.2.10/24;
      }
      mac 00:99:99:99:02:99;
    }
    unit 2 {
      family inet {
        address 192.0.2.11/24;
      }
      mac 00:99:99:99:03:99;
    }
    unit 10 {
      family inet {
        address 192.0.2.12/24;
      }
      mac 00:99:99:99:01:90;
    }
}
lo0 {
    unit 0 {
      family inet {
```

```
        address 198.51.100.1/24 {
          primary;
        }
      }
      family iso;
    }
}
```

```
user@PE1# show routing-options
router-id 198.51.100.1;
autonomous-system 65221;
forwarding-table {
  export load-balancing-policy;
}
```

```
user@PE1# show protocols
rsvp {
  interface all;
  interface fxp0.0 {
    disable;
  }
}
mpls {
  label-switched-path pe1tope2 {
    from 198.51.100.1;
    to 198.51.100.2;
    primary direct_to_pe2;
  }
  label-switched-path pe1tope3 {
    from 198.51.100.1;
    to 198.51.100.3;
    primary direct_to_pe3;
  }
  label-switched-path pe1tope4 {
    from 198.51.100.1;
    to 198.51.100.4;
    primary direct_to_pe4;
  }
  path pe4_to_pe3 {
    198.51.100.4 strict;
    198.51.100.3 strict;
  }
  path direct_to_pe2 {
    198.51.100.5 strict;
  }
  path direct_to_pe3 {
    198.51.100.6 strict;
  }
  path direct_to_pe4 {
    198.51.100.9 strict;
  }
  path pe2_to_pe3 {
    198.51.100.2 strict;
    198.51.100.3 strict;
```

```
    }
    interface all;
    interface fxp0.0 {
      disable;
    }
  }
  bgp {
    group RR {
      type internal;
      local-address 198.51.100.1;
      family evpn {
        signaling;
      }
      neighbor 203.0.113.0;
    }
  }
  isis {
    level 1 disable;
    interface all {
      level 2 metric 10;
    }
    interface fxp0.0 {
      disable;
    }
    interface lo0.0 {
      level 2 metric 0;
    }
  }
  ldp {
    deaggregate;
    interface all;
    interface fxp0.0 {
      disable;
    }
  }
  evpn {
  }
  l2-learning {
    global-mac-table-aging-time 18000;
  }
```

```
user@PE1# show policy-options
policy-statement load-balancing-policy {
    then {
      load-balance per-packet;
    }
}
```

```
user@PE1# show routing-instances
VS-1 {
    instance-type virtual-switch;
    interface ge-1/2/1.130;
    interface ge-1/2/2.120;
    interface ge-1/3/0.100;
```

```
    interface ae0.110;
    route-distinguisher 198.51.100.1:101;
    vrf-target target:100:101;
    protocols {
      evpn {
        extended-vlan-list [ 110 120 130 ];
        default-gateway do-not-advertise;
      }
    }
    bridge-domains {
      bd-110 {
        vlan-id 110;
        routing-interface irb.0;
      }
      bd-120 {
        vlan-id 120;
        routing-interface irb.1;
      }
      bd-130 {
        vlan-id 130;
        routing-interface irb.2;
      }
    }
}
VS-2 {
    instance-type virtual-switch;
    interface ge-1/2/1.230;
    interface ge-1/2/2.220;
    interface ge-1/3/0.200;
    interface ae0.210;
    route-distinguisher 198.51.100.1:201;
    vrf-target target:100:201;
    protocols {
      evpn {
        extended-vlan-list [ 210 220 230 ];
      }
    }
    bridge-domains {
      bd-a {
        vlan-id-list [ 210 220 230 ];
      }
    }
}
mhevpn {
    instance-type evpn;
    vlan-id 10;
    interface ge-1/2/1.0;
    interface ge-1/2/2.0;
    interface ge-1/3/0.0;
    interface ae0.0;
    routing-interface irb.10;
    route-distinguisher 198.51.100.1:1;
    vrf-target target:100:1;
    protocols {
      evpn {
```

```
        default-gateway do-not-advertise;
      }
    }
  }
  vrf {
    instance-type vrf;
    interface irb.0;
    interface irb.1;
    interface irb.2;
    interface irb.10;
    route-distinguisher 198.51.100.1:11;
    vrf-target target:100:11;
    vrf-table-label;
  }
```

Verification

Confirm that the configuration is working properly.

Verifying VPN Services in the Core

Purpose Ensure that the protocols in the VPN core are functioning properly.

Action From operational mode, enter the **show isis adjacency** command.

```
user@PE1> show isis adjacency

Interface             System         L State        Hold (secs) SNPA
ge-1/2/4.0                    CE10        2  Up                   24
5c:5e:ab:e:6f:4
```

From operational mode, enter the **show bgp summary** command.

```
user@PE1> show bgp summary

Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                      45         45          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
203.0.113.0              65221         90         26       0       0        3:18
 Establ
  bgp.evpn.0: 45/45/45/0
  VS-1.evpn.0: 19/19/19/0
  VS-2.evpn.0: 19/19/19/0
  mhevpn.evpn.0: 13/13/13/0
  __default_evpn__.evpn.0: 4/4/4/0
```

```
user@P> show bgp summary

Groups: 1 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                      68         68          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
198.51.100.1         65221         25         90       0       0        3:04
Establ
  bgp.evpn.0: 22/22/22/0
198.51.100.2         65221         32         80       0       0        6:12
Establ
  bgp.evpn.0: 22/22/22/0
198.51.100.3         65221         31         62       0       0        6:58
Establ
  bgp.evpn.0: 12/12/12/0
198.51.100.4         65221         28         88       0       0        6:04
Establ
  bgp.evpn.0: 12/12/12/0
```

From operational mode, enter the **show mpls lsp** command.

```
user@PE1> show mpls lsp

Ingress LSP: 3 sessions
To              From            State Rt P     ActivePath       LSPname
198.51.100.2   198.51.100.1   Up     0 *     direct_to_pe2    pe1tope2
198.51.100.3   198.51.100.1   Up     0 *     direct_to_pe3    pe1tope3
```

```
198.51.100.4   198.51.100.1   Up     0 *     direct_to_pe4    pe1tope4
Total 3 displayed, Up 3, Down 0

Egress LSP: 3 sessions
To              From            State   Rt Style Labelin Labelout LSPname
198.51.100.1   198.51.100.3   Up       0  1 FF       3        - pe3tope1
198.51.100.1   198.51.100.4   Up       0  1 FF       3        - pe4tope1
198.51.100.1   198.51.100.2   Up       0  1 FF       3        - pe2tope1
Total 3 displayed, Up 3, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

From operational mode, enter the **show interface ae* terse** command.

```
user@PE1> show interface ae* terse
```

```
Interface               Admin Link Proto    Local                 Remote
ae0                     up    up
ae0.0                   up    up   bridge
ae0.110                 up    up   bridge
ae0.210                 up    up   bridge
ae0.32767               up    up   multiservice
ae12                    up    up
ae12.0                  up    up   inet     198.51.100.12/24
                                   iso
                                   mpls
                                   multiservice
ae12.32767              up    up   multiservice
ae13                    up    up
ae13.0                  up    up   inet     198.51.100.13/24
                                   iso
                                   mpls
                                   multiservice
ae13.32767              up    up   multiservice
```

Meaning The protocols IS-IS, BGP and MPLS are up and running. The aggregated bundles configured on Router PE1 are up.

Verifying the EVPN Instance Status

Purpose Verify the EVPN routing instances and their status.

Action From operational mode, run the **show evpn instance extensive** command.

```
user@PE1> show evpn instance extensive
```

```
Instance: VS-1
  Route Distinguisher: 198.51.100.1:101
  Per-instance MAC route label: 301664
  MAC database status               Local  Remote
```

```
  Total MAC addresses:                0       0
  Default gateway MAC addresses:      3       0
Number of local interfaces: 4 (3 up)
  Interface name  ESI                             Mode             Status
  ae0.110         00:11:11:11:11:11:11:11:11:11   all-active       Up
  ge-1/2/1.130    00:33:33:33:33:33:33:33:33:33   all-active       Up
  ge-1/2/2.120    00:22:22:22:22:22:22:22:22:22   all-active       Up
  ge-1/3/0.100    00:00:00:00:00:00:00:00:00:00   single-homed     Up
Number of IRB interfaces: 3 (3 up)
  Interface name  VLAN ID  Status  L3 context
  irb.0           110      Up      vrf
  irb.1           120      Up      vrf
  irb.2           130      Up      vrf
Number of bridge domains: 3
  VLAN ID  Intfs / up    Mode             MAC sync  IM route label
  110          2   1     Extended         Enabled   301984
  120          2   1     Extended         Enabled   302000
  130          2   1     Extended         Enabled   302016
Number of neighbors: 3
  198.51.100.2
    Received routes
      MAC address advertisement:             0
      MAC+IP address advertisement:          0
      Inclusive multicast:                   3
      Ethernet auto-discovery:               6
  198.51.100.3
    Received routes
      MAC address advertisement:             0
      MAC+IP address advertisement:          0
      Inclusive multicast:                   1
      Ethernet auto-discovery:               2
  198.51.100.4
    Received routes
      MAC address advertisement:             0
      MAC+IP address advertisement:          0
      Inclusive multicast:                   3
      Ethernet auto-discovery:               2
Number of ethernet segments: 4
  ESI: 00:11:11:11:11:11:11:11:11:11
    Status: Resolved by IFL ae0.110
    Local interface: ae0.110, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.3    0          305584          all-active
      198.51.100.2    0          306000          all-active
    Designated forwarder: 198.51.100.3
    Backup forwarder: 198.51.100.1
    Backup forwarder: 198.51.100.2
    Advertised MAC label: 301792
    Advertised aliasing label: 301792
    Advertised split horizon label: 301808
  ESI: 00:22:22:22:22:22:22:22:22:22
    Status: Resolved by IFL ge-1/2/2.120
    Local interface: ge-1/2/2.120, Status: Up/Forwarding
    Number of remote PEs connected: 1
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.2    0          306032          all-active
    Designated forwarder: 198.51.100.1
    Backup forwarder: 198.51.100.2
```

```
    Advertised MAC label: 301824
    Advertised aliasing label: 301824
    Advertised split horizon label: 301840
  ESI: 00:33:33:33:33:33:33:33:33:33
    Status: Resolved by IFL ge-1/2/1.130
    Local interface: ge-1/2/1.130, Status: Up/Forwarding
    Number of remote PEs connected: 1
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.2    0          306064          all-active
    Designated forwarder: 198.51.100.1
    Backup forwarder: 198.51.100.2
    Advertised MAC label: 301856
    Advertised aliasing label: 301856
    Advertised split horizon label: 301872
  ESI: 00:44:44:44:44:44:44:44:44:44
    Status: Resolved by NH 1048613
    Number of remote PEs connected: 1
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.4    0          305152          all-active

Instance: VS-2
  Route Distinguisher: 198.51.100.1:201
  Per-instance MAC route label: 301696
  MAC database status               Local  Remote
    Total MAC addresses:                0       0
    Default gateway MAC addresses:      0       0
  Number of local interfaces: 4 (3 up)
    Interface name  ESI                             Mode             Status
    ae0.210         00:11:11:11:11:11:11:11:11:11   all-active       Up
    ge-1/2/1.230    00:33:33:33:33:33:33:33:33:33   all-active       Up
    ge-1/2/2.220    00:22:22:22:22:22:22:22:22:22   all-active       Up
    ge-1/3/0.200    00:00:00:00:00:00:00:00:00:00   single-homed     Down
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 3
    VLAN ID  Intfs / up    Mode             MAC sync  IM route label
    210          2   1     Extended         Enabled   302032
    220          2   1     Extended         Enabled   302048
    230          2   1     Extended         Enabled   302064
  Number of neighbors: 3
    198.51.100.2
      Received routes
        MAC address advertisement:             0
        MAC+IP address advertisement:          0
        Inclusive multicast:                   3
        Ethernet auto-discovery:               6
    198.51.100.3
      Received routes
        MAC address advertisement:             0
        MAC+IP address advertisement:          0
        Inclusive multicast:                   1
        Ethernet auto-discovery:               2
    198.51.100.4
      Received routes
        MAC address advertisement:             0
        MAC+IP address advertisement:          0
        Inclusive multicast:                   3
        Ethernet auto-discovery:               2
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
```

```
    Status: Resolved by IFL ae0.210
    Local interface: ae0.210, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.3    0          305648          all-active
      198.51.100.2    0          306096          all-active
    Designated forwarder: 198.51.100.1
    Backup forwarder: 198.51.100.2
    Backup forwarder: 198.51.100.3
    Advertised MAC label: 301888
    Advertised aliasing label: 301888
    Advertised split horizon label: 301808
  ESI: 00:22:22:22:22:22:22:22:22:22
    Status: Resolved by IFL ge-1/2/2.220
    Local interface: ge-1/2/2.220, Status: Up/Forwarding
    Number of remote PEs connected: 1
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.2    0          306112          all-active
    Designated forwarder: 198.51.100.1
    Backup forwarder: 198.51.100.2
    Advertised MAC label: 301904
    Advertised aliasing label: 301904
    Advertised split horizon label: 301840
  ESI: 00:33:33:33:33:33:33:33:33:33
    Status: Resolved by IFL ge-1/2/1.230
    Local interface: ge-1/2/1.230, Status: Up/Forwarding
    Number of remote PEs connected: 1
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.2    0          306128          all-active
    Designated forwarder: 198.51.100.1
    Backup forwarder: 198.51.100.2
    Advertised MAC label: 301920
    Advertised aliasing label: 301920
    Advertised split horizon label: 301872
  ESI: 00:44:44:44:44:44:44:44:44:44
    Status: Resolved by NH 1048616
    Number of remote PEs connected: 1
      Remote PE        MAC label  Aliasing label  Mode
      198.51.100.4    0          305184          all-active

Instance: __default_evpn__
  Route Distinguisher: 198.51.100.1:0
  VLAN ID: None
  Per-instance MAC route label: 301760
  MAC database status               Local  Remote
    Total MAC addresses:               0       0
    Default gateway MAC addresses:     0       0
  Number of local interfaces: 0 (0 up)
  Number of IRB interfaces: 0 (0 up)
  Number of neighbors: 2
    198.51.100.2
      Received routes
        Ethernet auto-discovery:              0
        Ethernet Segment:                     3
    198.51.100.3
      Received routes
        Ethernet auto-discovery:              0
        Ethernet Segment:                     1
```

```
  Number of ethernet segments: 0

Instance: mhevpn
  Route Distinguisher: 198.51.100.1:1
  VLAN ID: 10
  Per-instance MAC route label: 301728
  MAC database status               Local  Remote
    Total MAC addresses:               0       0
    Default gateway MAC addresses:     1       0
  Number of local interfaces: 4 (3 up)
    Interface name  ESI                            Mode             Status
    ae0.0           00:11:11:11:11:11:11:11:11:11  all-active       Up
    ge-1/2/1.0      00:33:33:33:33:33:33:33:33:33  all-active       Up
    ge-1/2/2.0      00:22:22:22:22:22:22:22:22:22  all-active       Up
    ge-1/3/0.0      00:00:00:00:00:00:00:00:00:00  single-homed     Down
  Number of IRB interfaces: 1 (1 up)
    Interface name  VLAN ID  Status  L3 context
    irb.10          10       Up      vrf
  Number of bridge domains: 1
    VLAN ID  Intfs / up    Mode             MAC sync  IM route label
    10           4   3     Extended         Enabled   302080
  Number of neighbors: 3
    198.51.100.2
      Received routes
        MAC address advertisement:             0
        MAC+IP address advertisement:          0
        Inclusive multicast:                   1
        Ethernet auto-discovery:               6
    198.51.100.3
      Received routes
        MAC address advertisement:             0
        MAC+IP address advertisement:          0
        Inclusive multicast:                   1
        Ethernet auto-discovery:               2
    198.51.100.4
      Received routes
        MAC address advertisement:             0
        MAC+IP address advertisement:          0
        Inclusive multicast:                   1
        Ethernet auto-discovery:               2
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Status: Resolved by IFL ae0.0
      Local interface: ae0.0, Status: Up/Forwarding
      Number of remote PEs connected: 2
        Remote PE        MAC label  Aliasing label  Mode
        198.51.100.3     0          305680          all-active
        198.51.100.2     0          306144          all-active
      Designated forwarder: 198.51.100.2
      Backup forwarder: 198.51.100.1
      Backup forwarder: 198.51.100.3
      Advertised MAC label: 301936
      Advertised aliasing label: 301936
      Advertised split horizon label: 301808
    ESI: 00:22:22:22:22:22:22:22:22:22
      Status: Resolved by IFL ge-1/2/2.0
      Local interface: ge-1/2/2.0, Status: Up/Forwarding
      Number of remote PEs connected: 1
        Remote PE        MAC label  Aliasing label  Mode
```

```
    198.51.100.2    0          306160          all-active
  Designated forwarder: 198.51.100.1
  Backup forwarder: 198.51.100.2
  Advertised MAC label: 301952
  Advertised aliasing label: 301952
  Advertised split horizon label: 301840
ESI: 00:33:33:33:33:33:33:33:33:33
  Status: Resolved by IFL ge-1/2/1.0
  Local interface: ge-1/2/1.0, Status: Up/Forwarding
  Number of remote PEs connected: 1
    Remote PE        MAC label  Aliasing label  Mode
    198.51.100.2    0          306176          all-active
  Designated forwarder: 198.51.100.1
  Backup forwarder: 198.51.100.2
  Advertised MAC label: 301968
  Advertised aliasing label: 301968
  Advertised split horizon label: 301872
ESI: 00:44:44:44:44:44:44:44:44:44
  Status: Resolved by NH 1048612
  Number of remote PEs connected: 1
    Remote PE        MAC label  Aliasing label  Mode
    198.51.100.4    0          305200          all-active
```

Meaning The output provides the following information:

- List of EVPN and virtual switch routing instances
- Mode of operation of each interface
- Neighbors of each routing instance
- Number of different routes received from each neighbor
- ESI attached to each routing instance
- Number of Ethernet segments on each routing instance
- DF election roles for each ESI in an EVI
- VLAN ID and MAC labels for each routing instance
- IRB interface details
- Number of default gateway MAC addresses received for the virtual switch routing instance (VS-1 and VS-2)

Verifying the Autodiscovery Routes per Ethernet Segment

Purpose Verify that the autodiscovery routes per Ethernet segment are received.

Action From operational mode, run the **show route table mhevpn.evpn.0** command.

Router PE1

```
user@PE1> show route table mhevpn.evpn.0
```

```
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:1::11111111111111111111::0/304
                   *[EVPN/170] 00:11:37
                      Indirect
1:198.51.100.1:1::22222222222222222222::0/304
                   *[EVPN/170] 00:11:37
                      Indirect
1:198.51.100.1:1::33333333333333333333::0/304
                   *[EVPN/170] 00:11:37
                      Indirect
1:198.51.100.2:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::22222222222222222222::FFFF:FFFF/304
                   *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::33333333333333333333::FFFF:FFFF/304
                   *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::11111111111111111111::0/304
                   *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::22222222222222222222::0/304
                   *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::33333333333333333333::0/304
                   *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.3:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 00:11:37, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
1:198.51.100.3:1::11111111111111111111::0/304
                   *[BGP/170] 00:11:37, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
3:198.51.100.1:1::10::198.51.100.1/304
                   *[EVPN/170] 00:13:38
                      Indirect
3:198.51.100.2:1::10::198.51.100.2/304
                   *[BGP/170] 00:11:33, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
3:198.51.100.3:1::10::198.51.100.3/304
                   *[BGP/170] 00:11:37, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
```

Router PE2

```
user@PE2> show route table mhevpn.evpn.0
```

```
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::22222222222222222222::FFFF:FFFF/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::33333333333333333333::FFFF:FFFF/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::11111111111111111111::0/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::22222222222222222222::0/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::33333333333333333333::0/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.2:1::11111111111111111111::0/304
                   *[EVPN/170] 01:10:26
                      Indirect
1:198.51.100.2:1::22222222222222222222::0/304
                   *[EVPN/170] 01:10:26
                      Indirect
1:198.51.100.2:1::33333333333333333333::0/304
                   *[EVPN/170] 01:10:26
                      Indirect
1:198.51.100.3:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.3:1::11111111111111111111::0/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.4:0::44444444444444444444::FFFF:FFFF/304
                   *[BGP/170] 01:10:17, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
1:198.51.100.4:1::44444444444444444444::0/304
                   *[BGP/170] 01:10:17, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
3:198.51.100.1:1::10::198.51.100.1/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
3:198.51.100.2:1::10::198.51.100.2/304
                   *[EVPN/170] 01:12:14
                      Indirect
```

```
3:198.51.100.3:1::10::198.51.100.3/304
                   *[BGP/170] 01:10:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
3:198.51.100.4:1::10::198.51.100.4/304
                   *[BGP/170] 01:10:17, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
```

Meaning The remote type 1 autodiscovery route is received for the ESI attached to Router PE2, which is the other PE router connected to the multihomed CE device.

Verifying the Ethernet Segment Route

Purpose Verify that the local and advertised autodiscovery routes per Ethernet segment and the Ethernet segment routes are received.

Action From operational mode, run the **show route table __default_evpn__.evpn.0** command.

Router PE1

```
user@PE1> show route table __default_evpn__.evpn.0

__default_evpn__.evpn.0: 10 destinations, 10 routes (10 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111111::FFFF:FFFF/304
                   *[EVPN/170] 00:25:18
                      Indirect
1:198.51.100.1:0::22222222222222222222::FFFF:FFFF/304
                   *[EVPN/170] 00:25:18
                      Indirect
1:198.51.100.1:0::33333333333333333333::FFFF:FFFF/304
                   *[EVPN/170] 00:25:18
                      Indirect
4:198.51.100.1:0::11111111111111111111:198.51.100.1/304
                   *[EVPN/170] 00:25:18
                      Indirect
4:198.51.100.1:0::22222222222222222222:198.51.100.1/304
                   *[EVPN/170] 00:25:18
                      Indirect
4:198.51.100.1:0::33333333333333333333:198.51.100.1/304
                   *[EVPN/170] 00:25:18
                      Indirect
4:198.51.100.2:0::11111111111111111111:198.51.100.2/304
                   *[BGP/170] 00:25:14, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
4:198.51.100.2:0::22222222222222222222:198.51.100.2/304
                   *[BGP/170] 00:25:14, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
```

```
4:198.51.100.2:0::33333333333333333333:198.51.100.2/304
                   *[BGP/170] 00:25:14, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
4:198.51.100.3:0::11111111111111111111:198.51.100.3/304
                   *[BGP/170] 00:25:18, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
```

Router PE2

```
user@PE2> show route table __default_evpn__.evpn.0
```

```
__default_evpn__.evpn.0: 10 destinations, 10 routes (10 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.2:0::11111111111111111111::FFFF:FFFF/304
                   *[EVPN/170] 01:17:59
                      Indirect
1:198.51.100.2:0::22222222222222222222::FFFF:FFFF/304
                   *[EVPN/170] 01:17:59
                      Indirect
1:198.51.100.2:0::33333333333333333333::FFFF:FFFF/304
                   *[EVPN/170] 01:17:59
                      Indirect
4:198.51.100.1:0::11111111111111111111:198.51.100.1/304
                   *[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
4:198.51.100.1:0::22222222222222222222:198.51.100.1/304
                   *[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
4:198.51.100.1:0::33333333333333333333:198.51.100.1/304
                   *[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
4:198.51.100.2:0::11111111111111111111:198.51.100.2/304
                   *[EVPN/170] 01:17:59
                      Indirect
4:198.51.100.2:0::22222222222222222222:198.51.100.2/304
                   *[EVPN/170] 01:17:59
                      Indirect
4:198.51.100.2:0::33333333333333333333:198.51.100.2/304
                   *[EVPN/170] 01:17:59
                      Indirect
4:198.51.100.3:0::11111111111111111111:198.51.100.3/304
                   *[BGP/170] 01:17:59, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
```

Meaning The output displays the local and remote type 1 (autodiscovery) and type 4 (Ethernet segment) routes.

Verifying the DF Status

Purpose Confirm which PE router is the designated forwarder (DF) for each routing instance.

Action From operational mode, run the **show evpn instance designated-forwarder** command.

```
user@PE1> show evpn instance designated forwarder

Instance: VS-1
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Designated forwarder: 198.51.100.3
    ESI: 00:22:22:22:22:22:22:22:22:22
      Designated forwarder: 198.51.100.1
    ESI: 00:33:33:33:33:33:33:33:33:33
      Designated forwarder: 198.51.100.1
    ESI: 00:44:44:44:44:44:44:44:44:44
      Designated forwarder: No local attachment to ethernet segment

Instance: VS-2
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Designated forwarder: 198.51.100.1
    ESI: 00:22:22:22:22:22:22:22:22:22
      Designated forwarder: 198.51.100.1
    ESI: 00:33:33:33:33:33:33:33:33:33
      Designated forwarder: 198.51.100.1
    ESI: 00:44:44:44:44:44:44:44:44:44
      Designated forwarder: No local attachment to ethernet segment

Instance: mhevpn
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Designated forwarder: 198.51.100.2
    ESI: 00:22:22:22:22:22:22:22:22:22
      Designated forwarder: 198.51.100.1
    ESI: 00:33:33:33:33:33:33:33:33:33
      Designated forwarder: 198.51.100.1
    ESI: 00:44:44:44:44:44:44:44:44:44
      Designated forwarder: No local attachment to ethernet segment
```

Meaning The designated forwarder is displayed for each routing instance and ESI.

Verifying the BDF Status

Purpose Confirm which PE router is the backup designated forwarder (BDF) for each routing instance.

Action From operational mode, run the **show evpn instance backup-forwarder** command.

```
user@PE1> show evpn instance backup-forwarder
```

```
Instance: VS-1
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Backup forwarder: 198.51.100.1
      Backup forwarder: 198.51.100.2
    ESI: 00:22:22:22:22:22:22:22:22:22
      Backup forwarder: 198.51.100.2
    ESI: 00:33:33:33:33:33:33:33:33:33
      Backup forwarder: 198.51.100.2
    ESI: 00:44:44:44:44:44:44:44:44:44
      Backup forwarder: No local attachment to ethernet segment

Instance: VS-2
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Backup forwarder: 198.51.100.2
      Backup forwarder: 198.51.100.3
    ESI: 00:22:22:22:22:22:22:22:22:22
      Backup forwarder: 198.51.100.2
    ESI: 00:33:33:33:33:33:33:33:33:33
      Backup forwarder: 198.51.100.2
    ESI: 00:44:44:44:44:44:44:44:44:44
      Backup forwarder: No local attachment to ethernet segment

Instance: mhevpn
  Number of ethernet segments: 4
    ESI: 00:11:11:11:11:11:11:11:11:11
      Backup forwarder: 198.51.100.1
      Backup forwarder: 198.51.100.3
    ESI: 00:22:22:22:22:22:22:22:22:22
      Backup forwarder: 198.51.100.2
    ESI: 00:33:33:33:33:33:33:33:33:33
      Backup forwarder: 198.51.100.2
    ESI: 00:44:44:44:44:44:44:44:44:44
      Backup forwarder: No local attachment to ethernet segment
```

Meaning The backup designated forwarder is displayed for each routing instance and ESI.

Verifying the Remote IRB and Host IP

Purpose Verify that the remote IRB IP and the host IP are received.

Router PE1

Action

From operational mode, run the **show route table mhevpn** command.

```
user@PE1> show route table mhevpn
```

```
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:1::11111111111111111111::0/304
                   *[EVPN/170] 01:34:26
```

```
                      Indirect
1:198.51.100.1:1::22222222222222222222::0/304
                    *[EVPN/170] 01:34:26
                      Indirect
1:198.51.100.1:1::33333333333333333333::0/304
                    *[EVPN/170] 01:34:26
                      Indirect
1:198.51.100.2:0::11111111111111111111::FFFF:FFFF/304
                    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::22222222222222222222::FFFF:FFFF/304
                    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:0::33333333333333333333::FFFF:FFFF/304
                    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::11111111111111111111::0/304
                    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::22222222222222222222::0/304
                    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.2:1::33333333333333333333::0/304
                    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
1:198.51.100.3:0::11111111111111111111::FFFF:FFFF/304
                    *[BGP/170] 01:34:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
1:198.51.100.3:1::11111111111111111111::0/304
                    *[BGP/170] 01:34:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
3:198.51.100.1:1::10::198.51.100.1/304
                    *[EVPN/170] 01:36:27
                      Indirect
3:198.51.100.2:1::10::198.51.100.2/304
                    *[BGP/170] 01:34:22, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.5 via ae12.0, label-switched-path pe1tope2
3:198.51.100.3:1::10::198.51.100.3/304
                    *[BGP/170] 01:34:26, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.6 via ae13.0, label-switched-path pe1tope3
```

Router PE2

From operational mode, run the **show route table mhevpn** command.

```
user@PE2> show route table mhevpn
```

```
mhevpn.evpn.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::22222222222222222222::FFFF:FFFF/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:0::33333333333333333333::FFFF:FFFF/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::11111111111111111111::0/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::22222222222222222222::0/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.1:1::33333333333333333333::0/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
1:198.51.100.2:1::11111111111111111111::0/304
                   *[EVPN/170] 01:35:11
                      Indirect
1:198.51.100.2:1::22222222222222222222::0/304
                   *[EVPN/170] 01:35:11
                      Indirect
1:198.51.100.2:1::33333333333333333333::0/304
                   *[EVPN/170] 01:35:11
                      Indirect
1:198.51.100.3:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.3:1::11111111111111111111::0/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
1:198.51.100.4:0::44444444444444444444::FFFF:FFFF/304
                   *[BGP/170] 01:35:02, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
1:198.51.100.4:1::44444444444444444444::0/304
                   *[BGP/170] 01:35:02, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
3:198.51.100.1:1::10::198.51.100.1/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.12 via ae12.0, label-switched-path pe2tope1
3:198.51.100.2:1::10::198.51.100.2/304
                   *[EVPN/170] 01:36:59
                      Indirect
```

```
3:198.51.100.3:1::10::198.51.100.3/304
                   *[BGP/170] 01:35:11, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.7 via ge-0/0/4.0, label-switched-path pe2tope3
3:198.51.100.4:1::10::198.51.100.4/304
                   *[BGP/170] 01:35:02, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.8 via ge-0/1/8.0, label-switched-path pe2tope4
```

Router PE3

From operational mode, run the **show route table mhevpn** command.

```
user@PE3> show route table mhevpn
```

```
mhevpn.evpn.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:0::22222222222222222222::FFFF:FFFF/304
                   *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:0::33333333333333333333::FFFF:FFFF/304
                   *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:1::11111111111111111111::0/304
                   *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:1::22222222222222222222::0/304
                   *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.1:1::33333333333333333333::0/304
                   *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
1:198.51.100.2:0::11111111111111111111::FFFF:FFFF/304
                   *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:0::22222222222222222222::FFFF:FFFF/304
                   *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:0::33333333333333333333::FFFF:FFFF/304
                   *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:1::11111111111111111111::0/304
                   *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
```

```
1:198.51.100.2:1::22222222222222222::0/304
                   *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.2:1::33333333333333333::0/304
                   *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
1:198.51.100.3:1::11111111111111111::0/304
                   *[EVPN/170] 01:36:25
                      Indirect
1:198.51.100.4:0::44444444444444444::FFFF:FFFF/304
                   *[BGP/170] 01:35:58, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.11 via ge-2/1/3.0, label-switched-path pe3tope4
1:198.51.100.4:1::44444444444444444::0/304
                   *[BGP/170] 01:35:58, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.11 via ge-2/1/3.0, label-switched-path pe3tope4
3:198.51.100.1:1::10::198.51.100.1/304
                   *[BGP/170] 01:36:10, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                     > to 198.51.100.13 via ae13.0, label-switched-path pe3tope1
3:198.51.100.2:1::10::198.51.100.2/304
                   *[BGP/170] 01:36:06, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.10 via ge-2/0/3.0, label-switched-path pe3tope2
3:198.51.100.3:1::10::198.51.100.3/304
                   *[EVPN/170] 01:37:33
                      Indirect
3:198.51.100.4:1::10::198.51.100.4/304
                   *[BGP/170] 01:35:58, localpref 100, from 203.0.113.0
                      AS path: I, validation-state: unverified
                    > to 198.51.100.11 via ge-2/1/3.0, label-switched-path pe3tope4
```

Meaning

The output displays the local and remote IRB interfaces. It also displays the local and remote hosts that are installed in the VRF table.

Release History Table

Release	Description
16.1R4	Starting with Junos OS Release 16.1R4, EVPN multihoming active-active mode is supported on all EX9200 switches.

Related Documentation

Example: Configuring LACP for EVPN Active-Active Multihoming

This example shows how to configure the Link Aggregation Control Protocol (LACP) on multihomed customer edge (CE) and provider edge (PE) devices in an Ethernet VPN (EVPN) active-active multihomed network.

Requirements

This example uses the following hardware and software components:

- Four MX Series 5G Universal Routing Platforms with MPC interfaces only, of which:
  - Two routers are configured as PE devices connected to a common multihomed customer site.
  - One router is configured as a remote PE device connected to a single-homed customer site.
  - One router is configured as the core provider device.
- Two CE devices, of which:
  - One CE device is multihomed to the two PE devices.
  - One CE device is single-homed to the remote PE device.
- Junos OS Release 17.1 or later running on all the PE devices.

Before you begin:

1. Configure the device interfaces and enable MPLS on all the interfaces of the PE devices.
2. Configure OSPF or any other IGP between the core provider device and the PE devices.
3. Configure MPLS and a label-switched path (LSP) from the ingress PE device to the remote egress PE device.
4. Configure an internal BGP session between the PE devices.

NOTE: We recommend that you configure Bidirectional Forwarding Detection (BFD) on the BGP session for faster detection of core isolation and better convergence.

Overview

Starting with Junos OS Release 17.1, an extra level of redundancy can be achieved in an EVPN active-active multihoming environment by configuring LACP on both the endpoints

of the multihomed CE-PE link. The multihomed devices are configured with aggregated trunk links, where the link aggregation group (LAG) interfaces of the CE-PE link can either be in the active or in the standby state. When the LAG interface is in the active state, data traffic is transmitted over the CE-PE link. When the LAG interface is in the standby state, data traffic is blocked and only control traffic for communicating LAG interface state is transmitted over the link.

The LAG interface state is monitored and operated by LACP to ensure fast convergence on isolation of a multihomed PE device from the core. When there is a core failure, a traffic black hole can occur at the isolated PE device. However, with the support for LACP on the CE-PE link, at the time of core isolation, the CE-facing interface of the multihomed PE device is set to the standby state, thereby blocking data traffic transmission from and toward the multihomed CE device. After the core recovers from the failure, the interface state is switched back from standby to active.

The support for LACP for EVPN active-active multihoming is applicable to both EVPN with MPLS and EVPN with VXLAN.

When LACP is configured on the CE-PE link, isolation of the multihomed PE device from the core is handled as follows:

1. When the EVPN BGP peers are null for a multihomed PE device, the PE device is isolated from the core. The CE devices connected to the isolated PE device are notified about the core failure by the isolated PE device.

2. On learning about the core failure, data traffic is not forwarded from the CE device to the isolated multihomed PE device. Instead, the traffic is diverted to the other multihomed PE devices that belong to the same LAG. This helps in preventing traffic black holes at the isolated PE device.

3. If the multihomed CE device uses the LAG for load balancing traffic to multiple active multihomed PE devices, then the LACP configuration along with the same system ID configured on all the multihomed PE devices for that given LAG, triggers an LACP out-of-sync to all the attached multihomed CE links.

When configuring LACP on the multihomed devices, be aware of the following considerations:

- The LAG link can operate either in the active or in the standby state regardless of the UP/DOWN operational state.
- On system reboot, the LACP link on the multihomed PE device is in the active state.
- When the control plane goes down or when the BGP-EVPN session is down, LACP is notified to run multiplexer state machine for the aggregation port and move the link from active to standby.
- An interface is not treated as up unless it operates in the active state and its operational state is also up.
- The following features are not supported with LACP on EVPN active-active multihoming:

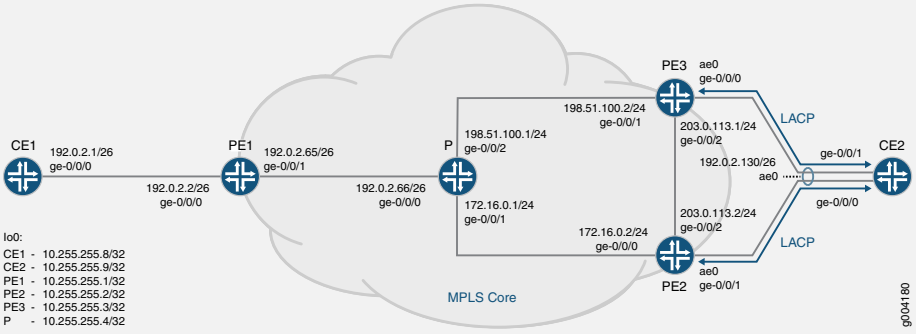
- Ethernet segment identifier (ESI) value auto-derivation from LACP system ID.
- Redundancy coverage for non-LAG interfaces.
- Redundancy coverage for core isolation with static LAGs.
- Node isolation.
- Access failures for LACP with EVPN active-active multihoming.

Topology

Figure 11 on page 175 illustrates an EVPN active-active multihoming network with LACP configured on the multihomed CE and PE devices. Device CE1 is single-homed and is connected to a remote PE device, PE1. Device PE2 and Device PE3 connect to a common multihomed CE—CE2. Device P is the core device to which all the PE devices (PE1, PE2, and PE3) are connected.

The endpoints of the multihomed devices—Device CE2, Device PE2, and Device PE3—are configured with LACP on the CE-PE link.

Figure 11: LACP Support in EVPN Active-Active Multihoming



Core isolation of Device PE3 is handled as follows:

1. After LACP is configured on the multihomed CE-PE links, the LACP configuration and operational data is synchronized between the LACP peers to operate as a LAG.

   The synchronization between the LACP peers happens with the exchange of control PDUs, and is required for the following reasons:

   - To determine the state of the links in the Ethernet bundle—all-active or standby.
   - To detect and handle CE device misconfiguration when LACP Port Key is configured on the PE device.
   - To detect and handle miswiring between CE and PE devices when LACP Port Key is configured on the PE device.
   - To detect and react to actor or partner churn when the LACP speakers are not able to converge.

2. After Device PE2 and Device PE3 establish BGP session with at least one peer, the state of the CE-PE link is set to unblocking mode by LACP.

3. When there is a core failure, and the BGP EVPN peers of Device PE2 becomes null, Device PE2 is isolated from the core. This can cause a traffic black hole at Device PE2. To prevent this situation, the LAG interface of Device PE2 that is facing Device CE2 is changed from the active state to the standby state by LACP.

4. An out-of-sync notification is sent from LACP on the attached multihomed CE2 link to block traffic transmission between Device CE2 and Device PE2.

5. When the control plane recovers, that is when Device PE2 establishes BGP session with other EVPN PEs, the LAG interface of Device PE2 is switched back from standby to active by LACP.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

Device CE1

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mac 00:00:00:00:00:01
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.1/26
set interfaces lo0 unit 0 family inet address 10.255.255.8/32
```

Device CE2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
```

```
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 mac 00:00:00:00:00:03
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 0 family inet address 192.0.2.130/26
set interfaces lo0 unit 0 family inet address 10.255.255.9/32
```

Device PE1

```
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/26
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 encapsulation flexible-ethernet-services
set interfaces ge-0/0/0 unit 0 encapsulation vlan-bridge
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.0.2.65/26
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.1/32
set routing-options router-id 10.255.255.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-0/0/1.0
set protocols bgp local-address 10.255.255.1
set protocols bgp family inet unicast
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.255.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.255.2
set protocols bgp group ibgp neighbor 10.255.255.3
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances ALPHA instance-type evpn
set routing-instances ALPHA vlan-id 100
set routing-instances ALPHA interface ge-0/0/0.0
set routing-instances ALPHA route-distinguisher 10.255.2:100
set routing-instances ALPHA vrf-target target:100:100
set routing-instances ALPHA protocols evpn label-allocation per-instance
```

Device PE2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 unit 0 family inet address 172.16.0.2/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 gigether-options 802.3ad ae0
set interfaces ge-0/0/2 unit 0 family inet address 203.0.113.2/24
set interfaces ge-0/0/2 unit 0 family iso
```

```
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:01:02:03:04:05
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
set interfaces lo0 unit 0 family inet address 10.255.255.2/32
set routing-options router-id 10.255.255.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 10.255.255.2
set protocols bgp family inet unicast
set protocols bgp family l2vpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.255.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.255.3
set protocols bgp group ibgp neighbor 10.255.255.1
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set protocols lacp ppm centralized
set routing-instances ALPHA instance-type evpn
set routing-instances ALPHA vlan-id 100
set routing-instances ALPHA interface ae0.0
set routing-instances ALPHA route-distinguisher 10.255.255.2:100
set routing-instances ALPHA vrf-target target:100:100
set switch-options service-id 1
```

Device PE3

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 gigether-options 802.3ad ae0
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.2/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 203.0.113.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ae0 vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:01:02:03:04:05
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 100
```

```
set interfaces lo0 unit 0 family inet address 10.255.255.3/32
set routing-options router-id 10.255.255.3
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 10.255.255.3
set protocols bgp family inet unicast
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.255.3
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.255.2
set protocols bgp group ibgp neighbor 10.255.255.1
set protocols bgp group l2vpn type internal
set protocols bgp group l2vpn family l2vpn signaling
set protocols bgp group l2vpn neighbor 10.255.255.4
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set protocols lacp ppm centralized
set routing-instances ALPHA instance-type evpn
set routing-instances ALPHA vlan-id 100
set routing-instances ALPHA interface ae0.0
set routing-instances ALPHA route-distinguisher 10.255.255.3:100
set routing-instances ALPHA vrf-target target:100:100
set switch-options service-id 1
```

Device P

```
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.66/26
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 172.16.0.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 198.51.100.1/24
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.4/32
set routing-options router-id 10.255.255.4
set routing-options autonomous-system 100
set protocols rsvp interface all aggregate
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/2.0
set protocols bgp group l2vpn type internal
set protocols bgp group l2vpn local-address 10.255.255.4
set protocols bgp group l2vpn family l2vpn signaling
set protocols bgp group l2vpn neighbor 10.255.255.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
```

```
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ldp interface all
set routing-instances vpls1 instance-type vpls
set routing-instances vpls1 route-distinguisher 10.255.255.4:100
set routing-instances vpls1 vrf-target target:200:200
set routing-instances vpls1 protocols vpls no-tunnel-services
set routing-instances vpls1 protocols vpls site vpls-pe site-identifier 3
set routing-instances vpls1 protocols vpls site vpls-pe best-site
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device PE2:

NOTE: Repeat this procedure for other multihomed PE devices after modifying the appropriate interface names, addresses, and other parameters.

1. Specify the number of aggregated Ethernet interfaces to be created on Device PE2.

```
[edit chassis]
user@PE2# set aggregated-devices ethernet device-count 1
```

2. Configure Device PE2 interfaces that connect to Device P and Device PE3, respectively.

```
[edit interfaces]
user@PE2# set ge-0/0/0 unit 0 family inet address 172.16.0.2/24
user@PE2# set ge-0/0/0 unit 0 family iso
user@PE2# set ge-0/0/0 unit 0 family mpls
user@PE2# set ge-0/0/2 unit 0 family inet address 203.0.113.2/24
user@PE2# set ge-0/0/2 unit 0 family iso
user@PE2# set ge-0/0/2 unit 0 family mpls
```

3. Configure Device PE2 interfaces within the ae0 aggregated bundle toward the multihomed Device CE2.

```
[edit interfaces]
user@PE2# set ge-0/0/1 gigether-options 802.3ad ae0
user@PE2# set ae0 vlan-tagging
user@PE2# set ae0 encapsulation flexible-ethernet-services
user@PE2# set ae0 unit 0 encapsulation vlan-bridge
user@PE2# set ae0 unit 0 vlan-id 100
```

4. Configure active-active multihoming on the aggregated Ethernet interface.

```
[edit interfaces]
```

```
user@PE2# set ae0 esi 00:11:22:33:44:55:66:77:88:99
user@PE2# set ae0 esi all-active
```

5. Configure LACP on the CE-PE link of Device PE2.

```
[edit interfaces]
user@PE2# set ae0 aggregated-ether-options lacp active
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:01:02:03:04:05
```

6. Configure the loopback interface of Device PE2.

```
[edit interfaces]
user@PE2# set lo0 unit 0 family inet address 10.255.255.2/32
```

7. Configure the router ID and autonomous system number for Device PE2.

```
[edit routing-options]
user@PE2# set router-id 10.255.255.2
user@PE2# set autonomous-system 100
```

8. Enable chained composite next hop for EVPN on Device PE2.

```
[edit routing-options]
user@PE2# set forwarding-table chained-composite-next-hop ingress evpn
```

9. Configure MPLS on all the interfaces of Device PE2, excluding the management interface.

```
[edit protocols]
user@PE2# set mpls interface all
user@PE2# set mpls interface fxp0.0 disable
```

10. Configure an internal BGP group for Device PE1 to peer with the other EVPN PE devices.

```
[edit protocols]
user@PE2# set bgp local-address 10.255.255.2
user@PE2# set bgp family inet unicast
user@PE2# set bgp family l2vpn signaling
user@PE2# set bgp group ibgp type internal
user@PE2# set bgp group ibgp local-address 10.255.255.2
user@PE2# set bgp group ibgp family evpn signaling
user@PE2# set bgp group ibgp neighbor 10.255.255.3
user@PE2# set bgp group ibgp neighbor 10.255.255.1
```

11. Configure OSPF and LDP on all the interfaces of Device PE2, excluding the management interface.

```
[edit protocols]
user@PE2# set ospf area 0.0.0.0 interface all
user@PE2# set ospf area 0.0.0.0 interface fxp0.0 disable
user@PE2# set ospf area 0.0.0.0 interface lo0.0 passive
user@PE2# set ldp interface all
user@PE2# set ldp interface fxp0.0 disable
user@PE2# set ldp interface lo0.0
```

12. Disable distributed PPM processing for only LACP packets.

```
[edit protocols]
user@PE2# set lacp ppm centralized
```

13. Configure an EVPN routing instance on Device PE2 and assign the routing instance parameters.

```
[edit routing-instances]
user@PE2# set ALPHA instance-type evpn
user@PE2# set ALPHA vlan-id 100
user@PE2# set ALPHA interface ae0.0
user@PE2# set ALPHA route-distinguisher 10.255.255.2:100
user@PE2# set ALPHA vrf-target target:100:100
```

14. Specify the service identifier for each multichassis aggregated Ethernet interface that belongs to the link aggregation group (LAG).

```
[edit switch-options]
user@PE2# set service-id 1
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show chassis
aggregated-devices {
   ethernet {
      device-count 1;
   }
}
```

```
user@PE2# show interfaces
ge-0/0/0 {
   unit 0 {
      family inet {
         address 172.16.0.2/24;
      }
      family iso;
```

```
      family mpls;
    }
  }
  ge-0/0/1 {
    gigether-options {
      802.3ad ae0;
    }
  }
  ge-0/0/2 {
    unit 0 {
      family inet {
        address 203.0.113.2/24;
      }
      family iso;
      family mpls;
    }
  }
  ae0 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:11:22:33:44:55:66:77:88:99;
      all-active;
    }
    aggregated-ether-options {
      lacp {
        active;
        system-id 00:01:02:03:04:05;
      }
    }
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id 100;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.255.2/32;
      }
    }
  }
```

```
user@PE2# show routing-options
router-id 10.255.255.2;
autonomous-system 100;
forwarding-table {
  chained-composite-next-hop {
    ingress {
      evpn;
    }
  }
}
```

```
user@PE2# show protocols
mpls {
   interface all;
   interface fxp0.0 {
      disable;
   }
}
bgp {
   local-address 10.255.255.2;
   family inet {
      unicast;
   }
   family l2vpn {
      signaling;
   }
   group ibgp {
      type internal;
      local-address 10.255.255.2;
      family evpn {
         signaling;
      }
      neighbor 10.255.255.3;
      neighbor 10.255.255.1;
   }
}
ospf {
   area 0.0.0.0 {
      interface all;
      interface fxp0.0 {
         disable;
      }
      interface lo0.0 {
         passive;
      }
   }
}
ldp {
   interface all;
   interface fxp0.0 {
      disable;
   }
   interface lo0.0;
}
lacp {
   ppm centralized;
}
```

```
user@PE2# show routing-instances
ALPHA {
   instance-type evpn;
   vlan-id 100;
   interface ae0.0;
   route-distinguisher 10.255.255.2:100;
   vrf-target target:100:100;
```

```
    protocols {
      evpn {
        traceoptions {
          file evpn-alpha.txt size 4294967295;
          flag all;
        }
      }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device CE2:

NOTE: Repeat this procedure for other multihomed CE devices after modifying the appropriate interface names, addresses, and other parameters.

1. Specify the number of aggregated Ethernet interfaces to be created on Device CE2.

```
[edit chassis]
user@CE2# set aggregated-devices ethernet device-count 1
```

2. Configure Device CE2 interfaces within the ae0 aggregated bundle toward Device PE2 and Device PE3.

```
[edit interfaces]
user@CE2# set ge-0/0/0 gigether-options 802.3ad ae0
user@CE2# set ge-0/0/1 gigether-options 802.3ad ae0
user@CE2# set ae0 flexible-vlan-tagging
user@CE2# set ae0 encapsulation flexible-ethernet-services
user@E2# set ae0 mac 00:00:00:00:00:03
user@E2# set ae0 unit 0 vlan-id 100
user@CE2# set ae0 unit 0 family inet address 192.0.2.130/26
```

3. Configure LACP on the CE-PE links on Device CE2.

```
[edit interfaces]
user@CE2# set ae0 aggregated-ether-options lacp active
```

4. Configure the loopback interface of Device CE2.

```
[edit interfaces]
user@CE2# set interfaces lo0 unit 0 family inet address 10.255.255.9/32
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE2# show chassis
aggregated-devices {
  ethernet {
    device-count 1;
  }
}
```

```
user@CE2# show interfaces
ge-0/0/0 {
  gigether-options {
    802.3ad ae0;
  }
}
ge-0/0/1 {
  gigether-options {
    802.3ad ae0;
  }
}
ae0 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  mac 00:00:00:00:00:03;
  aggregated-ether-options {
    lacp {
      active;
    }
  }
  unit 0 {
    vlan-id 100;
    family inet {
      address 192.0.2.130/26;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 10.255.255.9/32;
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying BGP Session Status

Purpose Verify that Device PE2 has established BGP peering with other EVPN PE devices.

Action From operational mode, run the **show bgp summary** command.

```
user@PE2> show bgp summary

Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
bgp.l2vpn.0
                       0          0          0          0          0          0
bgp.evpn.0
                       5          5          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
10.255.255.1            100       6316       6333       0       0 1d 23:33:03 Establ

  bgp.evpn.0: 1/1/1/0
  ALPHA.evpn.0: 1/1/1/0
  __default_evpn__.evpn.0: 0/0/0/0
10.255.255.3            100       6318       6332       0       0 1d 23:32:56 Establ

  bgp.evpn.0: 4/4/4/0
  ALPHA.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0
```

Meaning Device PE2 has established BGP peering with the other EVPN PE devices—Device PE1 and Device PE3.

Verifying LACP Interface Status of Multihomed PE Device

Purpose Verify the LACP interface state on Device PE2.

Action From operational mode, run the **show lacp interfaces** and **show interfaces ae* terse** commands.

```
user@PE2> show lacp interfaces

Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      ge-0/0/1       Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      ge-0/0/1     Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:        Receive State  Transmit State          Mux State
      ge-0/0/1                  Current   Fast periodic Collecting distributing
```

```
user@PE2> show interfaces ae* terse

Interface               Admin Link Proto    Local                 Remote
ae0                     up    up
ae0.0                   up    up   bridge
ae0.32767               up    up   multiservice
```

Meaning The LACP LAG interface state is active when there is BGP peering with other EVPN PE devices. The physical interface state of the aggregated Ethernet interfaces are up and operational.

Verifying LACP Interface State of Isolated PE Device

Purpose Verify the LACP interface state of Device PE2 when no BGP EVPN peers have been established.

Action From operational mode, run the **show lacp interfaces** and **show interfaces ae* terse** commands.

```
user@PE2> show lacp interfaces
```

```
Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      ge-0/0/1       Actor    No    No   No   No   No   Yes     Fast    Active
      ge-0/0/1     Partner    No    No   No   No   Yes  Yes     Fast    Active
    LACP protocol:        Receive State  Transmit State          Mux State
      ge-0/0/1                  Current   Fast periodic  Waiting
```

```
user@PE2> show interfaces ae* terse
```

```
Interface               Admin Link Proto    Local                 Remote
ae0                     up    down
ae0.0                   up    down   bridge
ae0.32767               up    down   multiservice
```

Meaning Because of core isolation, the LAG interface state is in standby, blocking traffic to-and-from Device CE2. As a result, the physical interface state of the aggregated Ethernet interface is also down until the LACP link state switches back to active on core failure recovery.

Verifying EVPN Routing Instance

Purpose Verify the EVPN routing instance parameters on Device PE2.

Action From operational mode, run the **show evpn instance extensive** command.

```
user@PE2> show evpn instance extensive

Instance: ALPHA
  Route Distinguisher: 10.255.255.2:100
  VLAN ID: 100
  Per-instance MAC route label: 299776
  MAC database status                       Local  Remote
    MAC advertisements:                         0       0
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 1 (1 up)
    Interface name  ESI                               Mode             Status
    ae0.0           00:11:22:33:44:55:66:77:88:99  all-active       Up
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label
    100                 1    1               Extended         Enabled   300048

  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES
    10.255.255.1            0         0         0         1         0
    10.255.255.3            0         0         2         1         0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      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
        10.255.255.3     0          299856          all-active
      Designated forwarder: 10.255.255.2
      Backup forwarder: 10.255.255.3
      Last designated forwarder update: Nov 21 01:27:52
      Advertised MAC label: 299856
      Advertised aliasing label: 299856
      Advertised split horizon label: 299968

Instance: __default_evpn__
  Route Distinguisher: 10.255.255.2:0
  Number of bridge domains: 0
  Number of neighbors: 1
    Address               MAC    MAC+IP        AD        IM        ES
    10.255.255.3            0         0         0         0         1
```

Meaning The output provides the following information:

- EVPN routing instance
- Mode of operation of each interface
- Neighbors of the routing instance
- Number of different routes received from each neighbor
- ESI attached to the routing instance

- Number of Ethernet segments on the routing instance
- Designated forwarder (DF) election roles for each ESI in an EVPN instance (EVI)
- VLAN ID and MAC labels for the routing instance

Related Documentation

- Example: Configuring EVPN Active-Active Multihoming on page 123

Example: Configuring LACP for EVPN VXLAN Active-Active Multihoming

This example shows how to configure the Link Aggregation Control Protocol (LACP) on multihomed customer edge (CE) and provider edge (PE) devices in an Ethernet VPN (EVPN) VXLAN active-active multihomed network.

Requirements

This example uses the following hardware and software components:

- Three QFX10002, QFX5100, QFX5110, QFX5200 switches, or QFX5100 Virtual Chassis configured as PE devices, and one QFX5100 switch configured as a CE device.
- Junos OS Release 17.1 or later running on all switches.

Overview

For another level of redundancy, you can configure EVPN VXLAN active-active multihoming by configuring LACP on both the endpoints of the multihomed CE-PE link. The multihomed devices are configured with aggregated trunk links, where the link aggregation group (LAG) interfaces of the CE-PE link can either be in the active or in the standby state. When the LAG interface is in the active state, data traffic is transmitted over the CE-PE link. When the LAG interface is in the standby state, data traffic is blocked and only control traffic for communicating LAG interface state is transmitted over the link.

LACP monitors and operates the LAG interface to ensure fast convergence on isolation of a multihomed PE device from the core. When there is a core failure, a traffic black hole can occur at the isolated PE device. However, with the support for LACP on the CE-PE link, at the time of core isolation, the CE-facing interface of the multihomed PE device is set to the standby state, thereby blocking data traffic transmission from and toward the multihomed PE device. After the core recovers from the failure, the interface state is switched back from standby to active.

NOTE: On QFX10002 and QFX10008 switches, only LACP for EVPN active-active multihoming with VXLAN is supported.

When you configure LACP on the CE-PE link, isolation of the multihomed PE device from the core is handled as follows:

1. The LACP peers synchronize the configuration and operational data.

   The LACP peers synchronize by exchanging control PDUs, and is required for the following reasons:

   - To determine the state of the links in the Ethernet bundle—all-active or standby.
   - To detect and handle CE device misconfiguration when LACP Port Key is configured on the PE device.
   - To detect and handle miswiring between CE and PE devices when LACP Port Key is configured on the PE device.
   - To detect and react to actor or partner churn when the LACP speakers are not able to converge.

2. When the peers are null for a multihomed PE device, the PE device is isolated from the core. In this case, the isolated PE device notifies the CE devices that are connected to the isolated PE device that there is a core failure.

3. Data traffic is not forwarded from the CE device to the isolated multihomed PE device. Instead, the traffic is diverted to the other multihomed PE devices that belong to the same LAG. This prevents traffic black holes at the isolated PE device.

4. If the multihomed CE device uses the LAG for load balancing traffic to multiple active multihomed PE devices, then the LACP configuration along with the same system ID configured on all the multihomed PE devices for that given LAG, triggers an LACP out-of-sync to all the attached multihomed CE links.

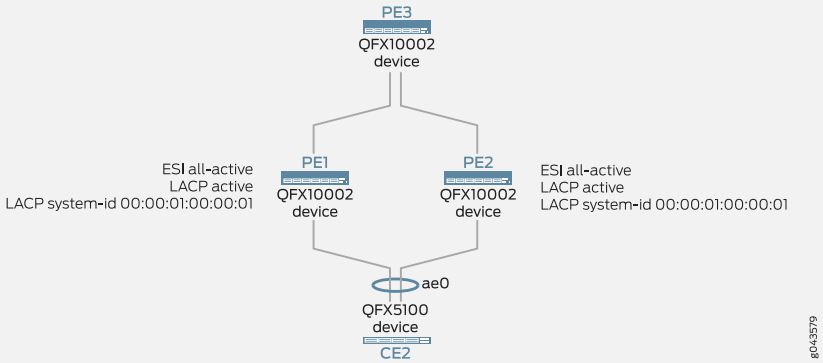
When configuring LACP on the multihomed devices, be aware of the following considerations:

- The LAG link can operate either in the active or in the standby state regardless of the UP/DOWN operational state.
- When you reboot the system, the LACP link on the multihomed PE device is in the active state.
- When the control plane goes down, LACP is notified to run multiplexer state machine for the aggregation port and move the link from active to standby.
- An interface is not treated as up unless it operates in the active state and its operational state is also up.

Topology

Figure 12 on page 193 illustrates an EVPN VXLAN active-active multihoming network with LACP configured on the multi-homed CE and PE devices. Device CE1 is single-homed and is connected to remote PE1 and PE2 devices. Device CE2 is multi-homed to PE1 and PE2 devices.

Figure 12: LACP Support in EVPN Active-Active Multihoming



Core isolation of Device PE1, for example, is handled as follows:

1. After PE2 and PE1 establish a BGP session, LACP sets the state of the CE-PE link to unblocking mode.

2. When there is a core failure, this can cause a traffic black hole at Device CE1.

   To prevent this situation, the LAG interface that is facing Device CE2 is changed from the active state to the standby state by LACP.

3. LACP sends an out of-sync notification on the attached multihomed CE2 link to block traffic transmission between Device CE2 and Device PE1.

4. When the control plane recovers, Device PE2 is switched back from standby to active by LACP.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

Device PE3

```
set interfaces xe-0/0/0 unit 0 family inet address 10.10.10.1/24
set interfaces xe-0/0/2 unit 0 family inet address 10.12.12.1/24
set interfaces xe-0/0/4 unit 0 family inet address 10.14.14.1/24
```

```
set interfaces xe-0/0/8 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/8 unit 0 family ethernet-switching vlan members all
set interfaces lo0 unit 0 family inet address 10.2.3.1/32 primary
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set vlans v100 vxlan vni 100
set vlans v200 vxlan vni 200
set routing-options router-id 10.2.3.1
set routing-options autonomous-system 11
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.2.3.1:1
set switch-options vrf-target target:1111:11
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.1
set protocols bgp group pe family evpn signaling
set protocols bgp group pe neighbor 10.2.3.3
set protocols bgp group pe neighbor 10.2.3.4
set protocols ospf area 0.0.0.0 interface xe-0/0/2
set protocols ospf area 0.0.0.0 interface lo0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0
set protocols ospf area 0.0.0.0 interface xe-0/0/4
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all
```

Device PE1

```
set chassis aggregated-devices ethernet device-count 1
set interfaces xe-0/0/2 unit 0 family inet address 10.12.12.2/24
set interfaces xe-0/0/3 unit 0 family inet address 10.11.11.2/24
set interfaces xe-0/0/9 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/9 unit 0 family ethernet-switching vlan members v200
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members all
set interfaces xe-0/0/55:0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:03:03:03:03:03:03:03:03:03
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
set interfaces lo0 unit 0 family inet address 10.2.3.3/32 primary
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set vlans v100 vxlan vni 100
set vlans v200 vxlan vni 200
set routing-options router-id 10.2.3.3
set routing-options autonomous-system 11
set switch-options vtep-source-interface lo0
set switch-options route-distinguisher 10.2.3.3:1
set switch-options vrf-target target:1111:11
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.3
set protocols bgp group pe family evpn signaling
set protocols bgp group pe neighbor 10.2.3.1
set protocols bgp group pe neighbor 10.2.3.4
set protocols ospf area 0.0.0.0 interface xe-0/0/2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

```
set protocols ospf area 0.0.0.0 interface xe-0/0/3
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all
```

Device PE2

```
set chassis aggregated-devices ethernet device-count 1
set interfaces xe-0/0/2 unit 0 family inet address 10.14.14.2/24
set interfaces xe-0/0/5 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/5 unit 0 family ethernet-switching vlan members all
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members all
set interfaces xe-0/0/55:0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:03:03:03:03:03:03:03:03:03
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
set interfaces lo0 unit 0 family inet address 10.2.3.4/32 primary
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set vlans v100 vxlan vni 100
set vlans v200 vxlan vni 200
set routing-options router-id 10.2.3.4
set routing-options autonomous-system 11
set switch-options vtep-source-interface lo0
set switch-options route-distinguisher 10.2.3.4:1
set switch-options vrf-target target:1111:11
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.4
set protocols bgp group pe family evpn signaling
set protocols bgp group pe neighbor 10.2.3.1
set protocols bgp group pe neighbor 10.2.3.2
set protocols bgp group pe neighbor 10.2.3.3
set protocols ospf area 0.0.0.0 interface xe-0/0/2
set protocols ospf area 0.0.0.0 interface lo0 passive
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all
```

Device CE2

```
set vlans v100 vlan-id 100
set vlans v200 vlan-id 200
set interfaces xe-0/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members all
set interfaces xe-0/0/5 ether-options 802.3ad ae0
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members all
set interfaces ae0 aggregated-ether-options lacp active
```

Configuring LACP for EVPN Active-Active Multihoming on PE3

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device PE3:

1. Configure uplink interfaces towards PE1 and PE2 devices.

```
[edit interfaces]
user@CE1# set xe-0/0/0 unit 0 family inet address 10.10.10.1/24
user@CE1# set xe-0/0/2 unit 0 family inet address 12.12.12.1/24
user@CE1# set xe-0/0/4 unit 0 family inet address 14.14.14.1/24
```

2. Configure xe-0/0/8 as a Layer 2 interface.

```
[edit interfaces]
user@CE1# set xe-0/0/8 unit 0 family ethernet-switching interface-mode trunk
user@CE1# set xe-0/0/8 unit 0 family ethernet-switching vlan members all
```

3. Configure a loopback interface.

```
[edit interfaces]
user@CE1# set lo0 unit 0 family inet address 10.2.3.1/32 primary
```

4. Create VLANs v100 and v200.

```
[edit vlans]
user@CE1# set v100 vlan-id 100
user@CE1# set v200 vlan-id 200
```

5. Map VLANs v100 and v200 to VNIs 100 and 200.

```
[edit vlans]
user@CE1# set v100 vxlan vni 100
user@CE1# set v200 vxlan vni 200
```

6. Configure the router ID and autonomous system number.

```
[edit routing-options]
user@CE1# set router-id 10.2.3.1
user@CE1# set autonomous-system 11
```

7. Specify the loopback interface as the source address for the VTEP tunnel.

```
[edit switch-options]
```

```
user@CE1# set vtep-source-interface lo0.0
```

8. Specify a route distinguisher to uniquely identify routes sent from this device.

```
[edit switch-options]
user@CE1# set route-distinguisher 10.2.3.1:1
```

9. Specify the global VRF export policy.

```
[edit switch-options]
user@CE1# set vrf-target target:1111:11
```

10. Configure an internal BGP group for PE3 to peer with PE1 and PE2.

```
[edit protocols]
user@CE1# set bgp group pe type internal
user@CE1# set bgp group pe local-address 10.2.3.1
user@CE1# set bgp group pe family evpn signaling
user@CE1# set bgp group pe neighbor 10.2.3.3
user@CE1# set bgp group pe neighbor 10.2.3.4
```

11. Configure an OSPF area.

```
[edit protocols]
user@CE1# set ospf area 0.0.0.0 interface xe-0/0/2
user@CE1# set ospf area 0.0.0.0 interface lo0 passive
user@CE1# set ospf area 0.0.0.0 interface xe-0/0/0
user@CE1# set ospf area 0.0.0.0 interface xe-0/0/4
```

12. Set VXLAN as the data plane encapsulation for EVPN.

```
[edit protocols]
user@CE1# set evpn encapsulation vxlan
```

13. Specify that all VNI(s) are advertised by EVPN.

```
[edit protocols]
user@CE1# set evpn extended-vni-list all
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show interfaces
```

```
xe-0/0/0 {
    unit 0 {
        family inet {
            address 10.10.10.1/24;
        }
    }
}
xe-0/0/2 {
    unit 0 {
        family inet {
            address 10.10.6.1/30;
            address 10.12.12.1/24;
        }
    }
}
xe-0/0/4 {
    unit 0 {
        family inet {
            address 10.14.14.1/24;
        }
    }
}
xe-0/0/8 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members all;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.10.0.1/32;
            address 10.2.3.1/32 {
                primary;
            }
        }
    }
}
```

```
user@PE3# show vlans
v100 {
    vlan-id 100;
    vxlan {
        vni 100;
    }
}
v200 {
    vlan-id 200;
    vxlan {
        vni 200;
```

```
    }
}
```

```
user@PE3# show routing-options
router-id 10.2.3.1;
autonomous-system 11;
```

```
user@PE3# show switching-options
vtep-source-interface lo0;
route-distinguisher 10.2.3.1:1;
vrf-target target:1111:11;
```

```
user@PE3# show protocols bgp
group pe {
    type internal;
    local-address 10.2.3.1;
    family evpn {
        signaling;
    }
    neighbor 10.2.3.3;
    neighbor 10.2.3.4;
}
```

```
user@PE3# show protocols ospf
area 0.0.0.0 {
    interface lo0.0 {
        passive;
    }
    interface xe-0/0/2;
    interface xe-0/0/0;
    interface xe-0/0/4;
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring LACP for EVPN Active-Active Multihoming on PE1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device PE1:

1. Specify the number of aggregated Ethernet interfaces to be created on Device PE1.

   ```
   [edit chassis]
   user@PE1# set aggregated-devices ethernet device-count 1
   ```

2. Configure the interfaces that connect to the CE device.

```
[edit interfaces]
user@PE1# set xe-0/0/2 unit 0 family inet address 10.12.12.2/24
user@PE1# set xe-0/0/2 unit 0 family inet address 10.11.11.2/24
```

3. Configure xe-0/0/9 as a Layer 2 interface.

```
[edit interfaces]
user@PE1# set xe-0/0/9 unit 0 family ethernet-switching interface-mode trunk
user@PE1# set xe-0/0/9 unit 0 family ethernet-switching vlan members v200
```

4. Configure ae0 as a Layer 2 interface.

```
[edit interfaces]
user@PE1# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@PE1# set ae0 unit 0 family ethernet-switching vlan members all
```

5. Configure the interface towards the multihomed device, CE2.

   Use the same ESI value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE1# set xe-0/0/55:0 ether-options 802.3ad ae0
user@PE1# set ae0 esi 00:03:03:03:03:03:03:03:03:03
user@PE1# set ae0 esi all-active
```

6. Configure LACP on the ae0.

   Use the same system ID value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE1# set ae0 aggregated-ether-options lacp active
user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
```

7. Configure a loopback interface.

```
[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 10.2.3.3/32 primary
```

8. Create VLANs v100 and v200.

```
[edit vlans]
user@PE1# set v100 vlan-id 100
user@PE1# set v200 vlan-id 200
```

9. Map VLANs v100 and v200 to VNIs 100 and 200.

```
[edit vlans]
```

```
user@PE1# set v100 vxlan vni 100
user@PE1# set v200 vxlan vni 200
```

10. Configure a router ID and autonomous system number.

```
[edit routing-options]
user@PE1# set router-id 10.2.3.3
user@PE1# set autonomous-system 11
```

11. Specify the loopback interface as the source address for the VTEP tunnel.

```
[edit switch-options]
user@PE1# set vtep-source-interface lo0.0
```

12. Specify a route distinguisher to uniquely identify routes sent from this device.

```
[edit switch-options]
user@PE1# set route-distinguisher 10.2.3.3:1
```

13. Specify the global VRF export policy.

```
[edit switch-options]
user@PE1# set vrf-target target:1111:11
```

14. Configure an internal BGP group for PE3 to peer with PE1 and PE2.

```
[edit protocols]
user@PE1# set bgp group pe type internal
user@PE1# set bgp group pe local-address 10.2.3.3
user@PE1# set bgp group pe family evpn signaling
user@PE1# set bgp group pe neighbor 10.2.3.1
user@PE1# set bgp group pe neighbor 10.2.3.4
```

15. Configure an OSPF area.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/2
user@PE1# set ospf area 0.0.0.0 interface lo0 passive
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/3
```

16. Set VXLAN as the data plane encapsulation for EVPN.

```
[edit protocols]
user@PE1# set evpn encapsulation vxlan
```

17. Specify that all VNI(s) are advertised by EVPN.

```
[edit protocols]
user@PE1# set evpn extended-vni-list all
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
   ethernet {
      device-count 1;
   }
}
```

```
user@PE1# show interfaces
xe-0/0/2 {
   unit 0 {
      family inet {
         address 10.10.6.1/30;
         address 10.12.12.1/24;
         address 10.12.12.2/24;
      }
   }
}
xe-0/0/3 {
   unit 0 {
      family inet {
         address 10.11.11.2/24;
      }
   }
}
xe-0/0/9 {
   unit 0 {
      family ethernet-switching {
         interface-mode trunk;
         vlan {
            members v200;
         }
      }
   }
}
xe-0/0/55:0 {
   ether-options {
      802.3ad ae0;
   }
}
ae0 {
   esi {
      00:03:03:03:03:03:03:03:03:03;
      all-active;
```



```
    }
    aggregated-ether-options {
      lacp {
        active;
        system-id 00:00:01:00:00:01;
      }
    }
    unit 0 {
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members all;
        }
      }
    }
}
```

```
user@PE1# show vlans
v100 {
    vlan-id 100;
    vxlan {
        vni 100;
    }
}
v200 {
    vlan-id 200;
    vxlan {
        vni 200;
    }
}
```

```
user@PE1# show routing-options
router-id 10.2.3.3;
autonomous-system 11;
```

```
user@PE1# show switching-options
vtep-source-interface lo0;
route-distinguisher 10.2.3.3:1;
vrf-target target:1111:11;
```

```
user@PE1# show protocols bgp
group pe {
    type internal;
    local-address 10.2.3.3;
    family evpn {
        signaling;
    }
    neighbor 10.2.3.3;
    neighbor 10.2.3.4;
    neighbor 10.2.3.1;
}
```

```
user@PE1# show protocols ospf
area 0.0.0.0 {
   interface lo0 {
      passive;
   }
   interface xe-0/0/2;
   interface xe-0/0/3;
}
```

```
user@PE1# show protocols evpn
encapsulation vxlan;
extended-vni-list all;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring LACP for EVPN Active-Active Multihoming on PE2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device PE2:

1. Specify the number of aggregated Ethernet interfaces to be created on Device PE1.

   ```
   [edit chassis]
   user@PE2# set aggregated-devices ethernet device-count 1
   ```

2. Configure the interface that connects to the CE device.

   ```
   [edit interfaces]
   user@PE2# set xe-0/0/2 unit 0 family inet address 10.14.14.2/24
   ```

3. Configure xe-0/0/5 as a Layer 2 interface.

   ```
   [edit interfaces]
   user@PE2# set xe-0/0/5 unit 0 family ethernet-switching interface-mode trunk
   user@PE2# set xe-0/0/5 unit 0 family ethernet-switching vlan members v200
   ```

4. Configure ae0 as a Layer 2 interface.

   ```
   [edit interfaces]
   user@PE2# set ae0 unit 0 family ethernet-switching interface-mode trunk
   user@PE2# set ae0 unit 0 family ethernet-switching vlan members all
   ```

5. Configure the interface towards the multihomed device, CE2.

   Use the same ESI value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE2# set xe-0/0/55:0 ether-options 802.3ad ae0
user@PE2# set ae0 esi 00:03:03:03:03:03:03:03:03:03
user@PE2# set ae0 esi all-active
```

6. Configure LACP on the ae0.

   Use the same system ID value on all PE devices where the CE2 is multihomed.

```
[edit interfaces]
user@PE2# set ae0 aggregated-ether-options lacp active
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:00:01:00:00:01
```

7. Configure a loopback interface.

```
[edit interfaces]
user@PE2# set lo0 unit 0 family inet address 10.2.3.4/32 primary
```

8. Create VLANs v100 and v200.

```
[edit vlans]
user@PE2# set v100 vlan-id 100
user@PE2# set v200 vlan-id 200
```

9. Map VLANs v100 and v200 to VNIs 100 and 200.

```
[edit vlans]
user@PE2# set v100 vxlan vni 100
user@PE2# set v200 vxlan vni 200
```

10. Configure a router ID and autonomous system number.

```
[edit routing-options]
user@PE2# set router-id 10.2.3.4
user@PE2# set autonomous-system 11
```

11. Specify the loopback interface as the source address for the VTEP tunnel.

```
[edit switch-options]
user@PE2# set vtep-source-interface lo0.0
```

12. Specify a route distinguisher to uniquely identify routes sent from this device.

```
[edit switch-options]
user@PE2# set route-distinguisher 10.2.3.4:1
```

13. Specify the global VRF export policy.

```
[edit switch-options]
user@PE2# set vrf-target target:1111:11
```

14. Configure an internal BGP group for PE3 to peer with PE1 and PE2.

```
[edit protocols]
user@PE2# set bgp group pe type internal
user@PE2# set bgp group pe local-address 10.2.3.4
user@PE2# set bgp group pe family evpn signaling
user@PE2# set bgp group pe neighbor 10.2.3.1
user@PE2# set bgp group pe neighbor 10.2.3.2
user@PE2# set bgp group pe neighbor 10.2.3.3
```

15. Configure an OSPF area.

```
[edit protocols]
user@PE2# set ospf area 0.0.0.0 interface xe-0/0/2
user@PE2# set ospf area 0.0.0.0 interface lo0 passive
```

16. Set VXLAN as the data plane encapsulation for EVPN.

```
[edit protocols]
user@PE2# set evpn encapsulation vxlan
```

17. Specify that all VNI(s) are advertised by EVPN.

```
[edit protocols]
user@PE2# set evpn extended-vni-list all
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show chassis
aggregated-devices {
   ethernet {
      device-count 1;
   }
}
```

```
user@PE2# show interfaces
xe-0/0/2 {
   unit 0 {
      family inet {
```

```
        address 10.14.14.2/24;
      }
    }
  }
  xe-0/0/5 {
    unit 0 {
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members all;
        }
      }
    }
  }
  xe-0/0/55:0 {
    ether-options {
      802.3ad ae0;
    }
  }
  ae0 {
    esi {
      00:03:03:03:03:03:03:03:03:03;
      all-active;
    }
    aggregated-ether-options {
      lacp {
        active;
        system-id 00:00:01:00:00:01;
      }
    }
    unit 0 {
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members all;
        }
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.2.3.4/32 {
          primary;
        }
      }
    }
  }
```

```
user@PE2# show vlans
v100 {
  vlan-id 100;
  vxlan {
    vni 100;
```

```
    }
}
v200 {
    vlan-id 200;
    vxlan {
        vni 200;
    }
}
```

```
user@PE2# show routing-options
router-id 10.2.3.4;
autonomous-system 11;
```

```
user@PE2# show switching-options
vtep-source-interface lo0;
route-distinguisher 10.2.3.4:1;
vrf-target target:1111:11;
```

```
user@PE2# show protocols bgp
group pe {
    type internal;
    local-address 10.2.3.4;
    family evpn {
        signaling;
    }
    neighbor 10.2.3.3;
    neighbor 10.2.3.4;
    neighbor 10.2.3.1;
}
```

```
user@PE2# show protocols ospf
area 0.0.0.0 {
    interface lo0 {
        passive;
    }
    interface xe-0/0/2;
}
```

```
user@PE2# show protocols evpn
encapsulation vxlan;
extended-vni-list all;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring LACP for EVPN Active-Active Multihoming on CE2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device CE1:

1. Configure VLANs v100 and v200.

```
[edit vlans]
user@CE2# set v100 vlan-id 100
user@CE2# set v200 vlan-id 200
```

2. Configure xe-0/0/3 as a Layer 2 interface.

```
[edit interfaces]
user@CE2# set xe-0/0/3 unit 0 family ethernet-switching interface-mode trunk
user@CE2# set xe-0/0/3 unit 0 family ethernet-switching vlan members all
```

3. Add member interfaces to ae0.

```
[edit interfaces]
user@CE2# set xe-0/0/0 ether-options 802.3ad ae0
user@CE2# set xe-0/0/5 ether-options 802.3ad ae0
```

4. Configure ae0 as a Layer 2 interface.

```
[edit interfaces]
user@CE2# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@CE2# set ae0 unit 0 family ethernet-switching vlan members all
```

5. Configure LACP as active for ae0.

```
[edit interfaces]
user@CE2# set ae0 aggregated-ether-options lacp active
```

Results

From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE2# show interfaces
xe-0/0/0 {
  ether-options {
    802.3ad ae0;
```

```
    }
  }
  xe-0/0/3 {
    unit 0 {
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members all;
        }
      }
    }
  }
```

```
user@CE2# show vlans
v100 {
  vlan-id 100;
  vxlan {
    vni 100;
  }
}
v200 {
  vlan-id 200;
  vxlan {
    vni 200;
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying LACP Interface Status of PE1 on page 210
- Verifying LACP Interface Status of PE2 on page 211

Verifying LACP Interface Status of PE1

Purpose Verify the LACP interface state on Device PE1.

Action From operational mode, run the **show lacp interfaces** command.

```
user@PE1> show lacp interfaces
```

```
Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      xe-0/0/55:0     Actor   No    No   Yes  Yes  Yes   Yes     Fast    Active
      xe-0/0/55:0     Partner No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:    Receive State  Transmit State      Mux State
      xe-0/0/55:0     Current        Fast periodic       Collecting distributing
```

Meaning The LACP LAG interface state is active.

Verifying LACP Interface Status of PE2

Purpose Verify the LACP interface state on Device PE2.

Action From operational mode, run the **show lacp interfaces** command.

```
user@PE2> show lacp interfaces
```

```
Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      xe-0/0/55:0     Actor   No    No   Yes  Yes  Yes   Yes     Fast    Acttive

      xe-0/0/55:0     Partner No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:    Receive State  Transmit State          Mux State
      xe-0/0/55:0     Current        Fast periodic           Collecting
distributing
```

Meaning The LACP LAG interface state is active.

Related Documentation

- *Understanding LACP for EVPN Active-Active Multihoming*

Configuring Dynamic List Next Hop

The routing table on a remote PE has a next hop entry for Ethernet segment identifier (ESI) routes with multiple next-hop elements for multihomed PE devices. For EVPN active-active multihoming device, the ESI route points to two next hop elements. Prior to dynamic list next hop, the routing protocol process (rpd) removed the next-hop entry for the ESI route when the link between the CE device and a multihome PE device goes down. The rpd would then create a new next hop entry for the ESI causing mass MAC route withdrawals and additions.

Starting in Junos OS Release 17.4R1, Junos OS supports the dynamic list next-hop feature in an EVPN network. Now when the link between the CE device and a multihomed PE device goes down, rather than removing the entire next hop and creating a new next hop for the ESI , the rpd removes the affected next hop element from the dynamic list next-hop entry for the ESI route. Dynamic list next hop provides the benefit of reducing mass MAC route withdrawals, improving the device performance, and reducing network convergence time.

To enable the dynamic list next-hop feature, include the **dynamic-list-next-hop** statement in the **[edit routing-options forwarding-table]** hierarchy.

NOTE: If you are performing an unified in-service software upgrade (ISSU) to upgrade your device from a Junos OS release prior to Junos OS Release 17.4R1, you must upgrade both the master Routing Engine and the backup Routing Engine before enabling the dynamic list next-hop feature.

To disable the dynamic list next-hop feature when it is enabled, use the **delete routing-options forwarding-table dynamic-list-next-hop** statement.

To display the next-hop elements from the Routing Engine's forwarding table, use the **show route label** and **show route forwarding-table** commands.

The following sample output from the **show route label detail** command shows two indirect next hops for an ESI with the dynamic list next-hop feature enabled.

```
user@host> show route label 299952 detail

mpls.0: 14 destinations, 14 routes (14 active, 0 holddown, 0 hidden)
299952 (1 entry, 1 announced)
TSI:
KRT in-kernel 299952 /52 -> {Dyn list:indirect(1048577), indirect(1048574)}
        *EVPN   Preference: 7
                Next hop type: Dynamic List, Next hop index: 1048575
                Address: 0x13f497fc
                Next-hop reference count: 5
                Next hop: ELNH Address 0xb7a3d90 uflags EVPN data
                    Next hop type: Indirect, Next hop index: 0
                    Address: 0xb7a3d90
                    Next-hop reference count: 3
                    Protocol next hop: 10.255.255.2
                    Label operation: Push 301344
                    Indirect next hop: 0x135b5c00 1048577 INH Session ID: 0x181
                        Next hop type: Router, Next hop index: 619
                        Address: 0xb7a3d30
                        Next-hop reference count: 4
                        Next hop: 1.0.0.4 via ge-0/0/1.0
                        Label operation: Push 301344, Push 299792(top)
                        Label TTL action: no-prop-ttl, no-prop-ttl(top)
                        Load balance label: Label 301344: None; Label 299792:
None;
                        Label element ptr: 0xb7a3cc0
                        Label parent element ptr: 0xb7a34e0
                        Label element references: 1
```

```
                        Label element child references: 0
                        Label element lsp id: 0
                Next hop: ELNH Address 0xb7a37f0 uflags EVPN data
                    Next hop type: Indirect, Next hop index: 0
                    Address: 0xb7a37f0
                    Next-hop reference count: 3
                    Protocol next hop: 10.255.255.3
                    Label operation: Push 301632
                    Indirect next hop: 0x135b5480 1048574 INH Session ID: 0x180
                        Next hop type: Router, Next hop index: 600
                        Address: 0xb7a3790
                        Next-hop reference count: 4
                        Next hop: 1.0.0.4 via ge-0/0/1.0
                        Label operation: Push 301632, Push 299776(top)
                        Label TTL action: no-prop-ttl, no-prop-ttl(top)
                        Load balance label: Label 301632: None; Label 299776:
None;
                        Label element ptr: 0xb7a3720
                        Label parent element ptr: 0xb7a3420
                        Label element references: 1
                        Label element child references: 0
                        Label element lsp id: 0
                State: <Active Int>
                Age: 1:18
                Validation State: unverified
                Task: evpn global task
                Announcement bits (2): 1-KRT 2-evpn global task
                AS path: I
                Routing Instance blue, Route Type Egress-MAC, ESI
00:11:22:33:44:55:66:77:88:99
```

The following sample output from the **show route forwarding table** command shows two next-hop entries for a destination with a multihomed route.

```
user@host> show route forwarding-table label 299952 extensive
```

```
MPLS:

Destination:  299952
  Route type: user
  Route reference: 0                    Route interface-index: 0
  Multicast RPF nh index: 0
  P2mpidx: 0
  Flags: sent to PFE, rt nh decoupled
  Next-hop type: indirect               Index: 1048575  Reference: 2
  Nexthop:
  Next-hop type: composite              Index: 601      Reference: 2
  Next-hop type: indirect               Index: 1048574  Reference: 3
  Nexthop: 1.0.0.4
  Next-hop type: Push 301632, Push 299776(top) Index: 600 Reference: 2
  Load Balance Label: None
  Next-hop interface: ge-0/0/1.0
  Next-hop type: indirect               Index: 1048577  Reference: 3
  Nexthop: 1.0.0.4
  Next-hop type: Push 301344, Push 299792(top) Index: 619 Reference: 2
  Load Balance Label: None
  Next-hop interface: ge-0/0/1.0
```

The following sample shows the **show route forwarding table** command output after one of the PE devices has been disabled. It shows one next-hop element and one empty next-hop element.

```
user@host> show route forwarding-table label 299952 extensive

Routing table: default.mpls [Index 0]
MPLS:

Destination:  299952
  Route type: user
  Route reference: 0                    Route interface-index: 0
  Multicast RPF nh index: 0
  P2mpidx: 0
  Flags: sent to PFE, rt nh decoupled
  Next-hop type: indirect               Index: 1048575  Reference: 2
  Nexthop:
  Next-hop type: composite              Index: 601      Reference: 2
  Next-hop type: indirect               Index: 1048577  Reference: 3
  Nexthop: 1.0.0.4
  Next-hop type: Push 301344, Push 299792(top) Index: 619 Reference: 2
  Load Balance Label: None
  Next-hop interface: ge-0/0/1.0
```

Example: Configuring an ESI on a Logical Interface With EVPN Multihoming

When a customer edge (CE) device in an Ethernet VPN-Multiprotocol Label Switching (EVPN-MPLS) environment is multihomed to two or more provider edge (PE) devices, the set of Ethernet links that connect the devices comprise an Ethernet segment. An Ethernet segment identifier (ESI) is a 10-octet integer that identifies this segment. A sample ESI is 00:11:22:33:44:55:66:77:88:99.

In releases before Junos OS Release 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI only on a physical or aggregated Ethernet interface, for example, **set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99**. If you specify an ESI on a physical or aggregated Ethernet interface, keep in mind that an ESI is a factor in the designated forwarder (DF) election process. For example, assume that you configure EVPN multihoming active-standby on aggregated Ethernet interface ae0, and given the ESI configured on ae0 and other determining factors, the DF election results in ae0 being in the down state. Further, all logical interfaces configured on ae0, for example, **set interfaces ae0 unit 1** and **set interfaces ae0 unit 2** are also in the down state, which renders logical interfaces ae0.1 and ae0.2 unable to provide services to their respective customer sites (VLANs).

Starting with Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. If you specify an ESI on a logical interface, the DF election process now occurs at the individual logical interface level, which enables you to better utilize logical interfaces. For example, assume that you configure logical interfaces ae0.1 and ae0.2 on aggregated Ethernet interface ae0. You configure EVPN multihoming active-standby on both logical interfaces and given the ESI configured on ae0.1 and other determining factors, the DF election results in ae0.1 being in the down state. Despite logical interface ae0.1 being down, logical

interface ae0.2 and other logical interfaces configured on ae0 can be in the up state and provide services to their respective customer sites (VLANs).

This topic shows how to configure an ESI on logical interfaces in both EVPN multihoming active-standby and active-active modes.

Requirements

Both EVPN multihoming active-standby and multihoming active-active examples use the following hardware and software components:

- An EX9200 switch running Junos OS Release 17.3R1 or later (PE1)
- An MX Series router running Junos OS Release 15.1F6 or later, or Junos OS Release 17.1R1 or later (PE2)

Overview and Topology

EVPN Multihoming Active-Standby

Figure 13 on page 215 shows an EVPN-MPLS topology in which CE1 is multihomed to PE1 and PE2 to provide redundant paths to CE2. On CE1, the connections to PE1 and PE2 are configured as separate aggregated Ethernet interfaces. Table 6 on page 216 shows how the connections with CE1 are configured on PE1 and PE2. Note that the EVPN multihoming mode, ESI, and VLAN ID are actually configured on logical interfaces ae0.1 on each PE device.

Figure 13: EVPN-MPLS Topology with Multihoming Active-Standby

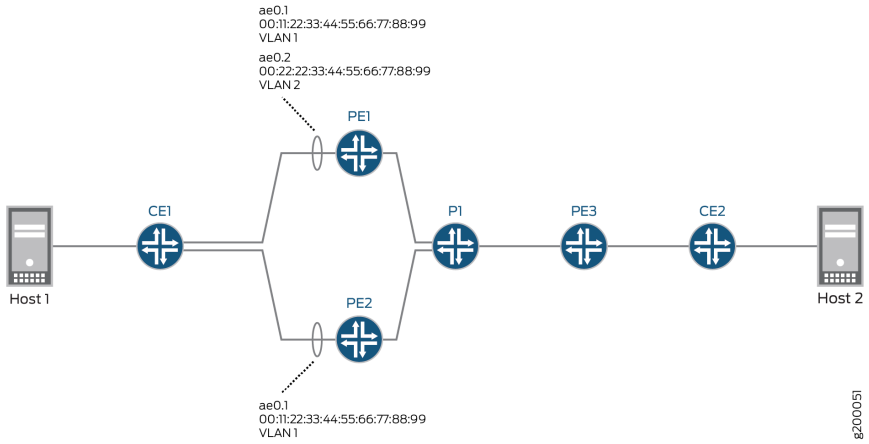


Table 6: EVPN Multihoming Active-Standby: Configuring the Connection with CE1 on PE1 and PE2

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.1	single-active	00:11:22:33:44:55:66:77:88:99	1
PE2	xe-3/0/2	ae0	ae0.1	single-active	00:11:22:33:44:55:66:77:88:99	1

Based on the DF election, logical interface ae0.1 on PE2 is up, and logical interface ae0.1 on PE1 is down.

Table 7 on page 216 also shows the configuration for logical interface ae0.2 on PE1. Note that logical interface ae0.2 provides services for a different VLAN and is configured with a different ESI than logical interface ae0.1, which is configured on the same aggregated Ethernet interface. As a result, logical interface ae0.2 is up and providing services to VLAN 2 despite the fact that logical interface ae0.1 is in the down state.

Table 7: Multihoming Active-Standby: Configuring Logical Interface on PE1 that Provides Services to a Different VLAN

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.2	single-active	00:22:33:44:55:66:77:88:99	2

EVPN Multihoming Active-Active

Figure 14 on page 216 shows an EVPN-MPLS topology in which CE1 is multihomed to PE1 and PE2 to provide redundant paths to CE2. On CE1, the connections to PE1 and PE2 are configured as one aggregated Ethernet interface. Table 8 on page 217 shows how the connections with CE1 are configured on PE1 and PE2. Note that the EVPN multihoming mode, ESI, and VLAN ID are actually configured on logical interfaces ae0.1 on each device.

Figure 14: EVPN-MPLS Topology with Multihoming Active-Active

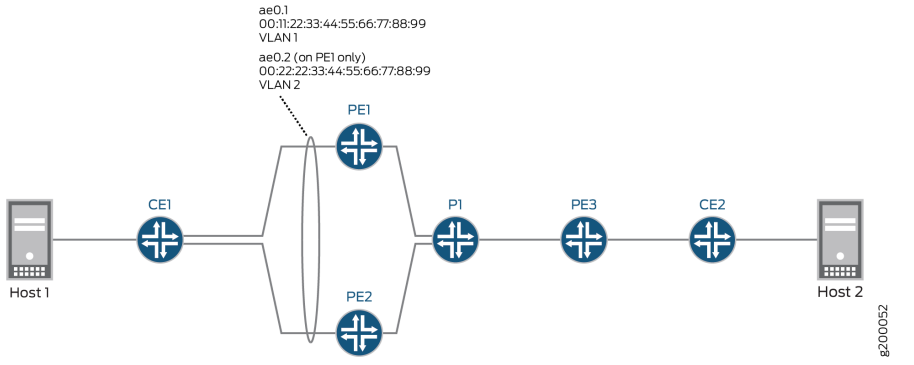


Table 8: Multihoming Active-Active: Configuring the Connection with CE1 on PE1 and PE2

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.1	all-active	00:11:22:33:44:55:66:77:88:99	1
PE2	xe-3/0/2	ae0	ae0.1	all-active	00:11:22:33:44:55:66:77:88:99	1

Based on the DF election, logical interface ae0.1 on PE1 is in the up state, and logical interface ae0.1 on PE2 is in the down state.

Table 9 on page 217 also shows the configuration for logical interface ae0.2 on PE1. Note that logical interface ae0.2 provides services for a different VLAN and is configured with a different ESI than logical interface ae0.1, which is in the same aggregated Ethernet interface. As a result, logical interface ae0.2 is down and unable to provide services to VLAN 2 despite the fact that logical interface ae0.1 is in the up state.

Table 9: EVPN Multihoming Active-Active: Configuring Interface on PE1 that Provides Services to a Different VLAN

Device	Physical Interface	Aggregated Ethernet Interface	Logical Interface	EVPN Multihoming Mode	ESI	VLAN ID
PE1	xe-2/0/0	ae0	ae0.2	all-active	00:22:22:33:44:55:66:77:88:99	2

EVPN Multihoming Active-Standby Configuration

NOTE: The configurations for PE1 (EX9200) and PE2 (MX Series router) focus on configuring EVPN multhihoming active-standby and ESIs on logical interfaces. The configurations do not include all EVPN-related configurations for physical interfaces, aggregated Ethernet interfaces, logical interfaces, and routing instances. For a more comprehensive configuration of EVPN multihoming active-standby in an EVPN-MPLS environment, see "Example: Configuring EVPN Active-Standby Multihoming" on page 79. Note that the referenced example shows how to configure ESIs on physical and aggregated Ethernet interfaces.

CLI Quick Configuration

PE1

```
set interfaces xe-2/0/0 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi single-active
set interfaces ae0 unit 2 encapsulation vlan-bridge
set interfaces ae0 unit 2 vlan-id 2
set interfaces ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
```

```
set interfaces ae0 unit 2 esi single-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue l3-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
set routing-instances green instance-type evpn
set routing-instances green vlan-id 2
set routing-instances green interface ae0.2
set routing-instances green l3-interface irb.2
...
set routing-instances green protocols evpn interface ae0.2
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
...
```

PE2

```
set interfaces xe-3/0/2 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi single-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue routing-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.1
...
```

Step-by-Step Procedure

To configure EVPN multihoming active-standby on PE1:

1. Specify an Ethernet interface as a member of aggregated Ethernet interface ae0.

   ```
   [edit interfaces]
   user@switch# set xe-2/0/0 gigether-options 802.3ad ae0
   ```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

   ```
   [edit interfaces]
   ```

```
user@switch# set ae0 flexible-vlan-tagging
user@switch# set ae0 encapsulation flexible-ethernet-services
```

3. On aggregated Ethernet interface ae0, configure logical interfaces ae0.1 and ae0.2. Configure the logical interfaces to use VLAN bridge encapsulation, and map the logical interfaces to VLANs 1 and 2, respectively. Also, assign an ESI to the logical interfaces, and enable EVPN multihoming active-standby.

```
[edit interfaces]
user@switch# set ae0 unit 1 encapsulation vlan-bridge
user@switch# set ae0 unit 1 vlan-id 1
user@switch# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set ae0 unit 1 esi single-active
user@switch# set ae0 unit 2 encapsulation vlan-bridge
user@switch# set ae0 unit 2 vlan-id 2
user@switch# set ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
user@switch# set ae0 unit 2 esi single-active
```

4. Configure IRB interfaces irb.1 and irb.2, and assign an IP address to each interface.

```
[edit interfaces]
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure EVPN logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set blue instance-type evpn
user@switch# set blue vlan-id 1
user@switch# set blue interface ae0.1
user@switch# set blue l3-interface irb.1
user@switch# set blue protocols evpn interface ae0.1
```

6. Configure an EVPN routing instance named green. Map the routing instance to VLAN 2, logical interface ae0.2, and IRB interface irb.2. Configure logical interface ae0.2 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set green instance-type evpn
user@switch# set green vlan-id 2
user@switch# set green interface ae0.2
user@switch# set green l3-interface irb.2
user@switch# set green protocols evpn interface ae0.2
```

7. Configure a VRF routing instance, and add IRB interfaces irb.1 and irb.2 to the routing instance.

```
[edit routing-instances]
set vrf instance-type vrf
set vrf interface irb.1
set vrf interface irb.2
```

Step-by-Step Procedure

To configure EVPN multihoming active-standby on PE2:

1. Specify an Ethernet interface as a member of aggregated Ethernet interface ae0.

```
[edit interfaces]
user@router# set xe-3/0/2 gigether-options 802.3ad ae0
```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

```
[edit interfaces]
user@router# set ae0 flexible-vlan-tagging
user@router# set ae0 encapsulation flexible-ethernet-services
```

3. On aggregated Ethernet interface ae0, configure logical interface ae0.1. Configure the logical interface to use VLAN bridge encapsulation, and map the logical interface to VLAN 1 and 2. Also, assign an ESI to the logical interface, and enable EVPN multihoming active-standby.

```
[edit interfaces]
user@router# set ae0 unit 1 encapsulation vlan-bridge
user@router# set ae0 unit 1 vlan-id 1
user@router# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@router# set ae0 unit 1 esi single-active
```

4. Configure IRB interface irb.1, and assign an IP address to the interface.

```
[edit interfaces]
user@router# set irb unit 1 family inet address 192.0.2.1/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure EVPN logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@router# set blue instance-type evpn
user@router# set blue vlan-id 1
```

```
user@router# set blue interface ae0.1
user@router# set blue routing-interface irb.1
user@router# set blue protocols evpn interface ae0.1
```

6. Configure a VRF routing instance, and add IRB interface irb.1 to the routing instance.

```
[edit routing-instances]
user@router# set vrf instance-type vrf
user@router# set vrf interface irb.1
```

EVPN Multihoming Active-Active Configuration

CLI Quick Configuration

NOTE: The configurations for PE1 (EX9200) and PE2 (MX Series router) focus on configuring EVPN multihoming active-active and ESIs on logical interfaces. The configurations do not include all EVPN-related configurations for physical interfaces, aggregated Ethernet interfaces, logical interfaces, and routing instances. For a more comprehensive configuration of EVPN multihoming active-active, see "Example: Configuring EVPN Active-Active Multihoming" on page 123. Note that the referenced example shows how to configure ESIs on physical and aggregated Ethernet interfaces.

PE1

```
set interfaces xe-2/0/0 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces ae0 unit 2 encapsulation vlan-bridge
set interfaces ae0 unit 2 vlan-id 2
set interfaces ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
set interfaces ae0 unit 2 esi all-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue l3-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
set routing-instances green instance-type evpn
set routing-instances green vlan-id 2
set routing-instances green interface ae0.2
set routing-instances green l3-interface irb.2
...
set routing-instances green protocols evpn interface ae0.2
```

```
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.1
set routing-instances vrf interface irb.2
...
```

PE2

```
set interfaces xe-3/0/2 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 unit 1 encapsulation vlan-bridge
set interfaces ae0 unit 1 vlan-id 1
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces irb unit 1 family inet address 192.0.2.1/24
set routing-instances blue instance-type evpn
set routing-instances blue vlan-id 1
set routing-instances blue interface ae0.1
set routing-instances blue routing-interface irb.1
...
set routing-instances blue protocols evpn interface ae0.1
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.1
...
```

Step-by-Step Procedure

To configure EVPN multihoming active-active on PE1:

1. Specify an Ethernet interface as a member of aggregated Ethernet interface ae0.

   ```
   [edit interfaces]
   user@switch# set xe-2/0/0 gigether-options 802.3ad ae0
   ```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

   ```
   [edit interfaces]
   user@switch# set ae0 flexible-vlan-tagging
   user@switch# set ae0 encapsulation flexible-ethernet-services
   ```

3. On aggregated Ethernet interface ae0, configure logical interfaces ae0.1 and ae0.2. Configure the logical interfaces to use VLAN bridge encapsulation, and map the logical interfaces to VLANs 1 and 2, respectively. Also, assign an ESI to the logical interfaces, and enable EVPN multihoming active-active.

   ```
   [edit interfaces]
   user@switch# set ae0 unit 1 encapsulation vlan-bridge
   user@switch# set ae0 unit 1 vlan-id 1
   user@switch# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
   user@switch# set ae0 unit 1 esi all-active
   user@switch# set ae0 unit 2 encapsulation vlan-bridge
   ```

```
user@switch# set ae0 unit 2 vlan-id 2
user@switch# set ae0 unit 2 esi 00:22:22:33:44:55:66:77:88:99
user@switch# set ae0 unit 2 esi all-active
```

4. Configure IRB interfaces irb.1 and irb.2, and assign an IP address to each interface.

```
[edit interfaces]
set interfaces irb unit 1 family inet address 192.0.2.1/24
set interfaces irb unit 2 family inet address 192.0.2.2/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set blue instance-type evpn
user@switch# set blue vlan-id 1
user@switch# set blue interface ae0.1
user@switch# set blue l3-interface irb.1
user@switch# set blue protocols evpn interface ae0.1
```

6. Configure an EVPN routing instance named green. Map the routing instance to VLAN 2, logical interface ae0.2, and IRB interface irb.2. Configure logical interface ae0.2 for the EVPN routing instance.

```
[edit routing-instances]
user@switch# set green instance-type evpn
user@switch# set green vlan-id 2
user@switch# set green interface ae0.2
user@switch# set green l3-interface irb.2
user@switch# set green protocols evpn interface ae0.2
```

7. Configure a VRF routing instance, and add IRB interfaces irb.1 and irb.2 to the routing instance.

```
[edit routing-instances]
set vrf instance-type vrf
set vrf interface irb.1
set vrf interface irb.2
```

Step-by-Step Procedure

To configure EVPN multihoming active-active on PE2:

1. Specify Ethernet interface xe-3/0/2 as a member of aggregated Ethernet interface ae0.

```
[edit interfaces]
```

```
user@router# set xe-3/0/2 gigether-options 802.3ad ae0
```

2. Configure aggregated Ethernet interface ae0 to simultaneously transmit 802.1Q VLAN single-tag and dual-tag frames and to support different types of Ethernet encapsulation at the logical interface level.

```
[edit interfaces]
user@router# set ae0 flexible-vlan-tagging
user@router# set ae0 encapsulation flexible-ethernet-services
```

3. On aggregated Ethernet interface ae0, configure logical interface ae0.1. Configure the logical interface to use VLAN bridge encapsulation, and map the logical interface to VLANs 1. Also, assign an ESI to the logical interface, and enable EVPN multihoming active-active.

```
[edit interfaces]
user@router# set ae0 unit 1 encapsulation vlan-bridge
user@router# set ae0 unit 1 vlan-id 1
user@router# set ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@router# set ae0 unit 1 esi all-active
```

4. Configure IRB interface irb.1, and assign an IP address to the interface.

```
[edit interfaces]
user@router# set irb unit 1 family inet address 192.0.2.1/24
```

5. Configure an EVPN routing instance named blue. Map the routing instance to VLAN 1, logical interface ae0.1, and IRB interface irb.1. Configure logical interface ae0.1 for the EVPN routing instance.

```
[edit routing-instances]
user@router# set blue instance-type evpn
user@router# set blue vlan-id 1
user@router# set blue interface ae0.1
user@router# set blue routing-interface irb.1
user@router# set blue protocols evpn interface ae0.1
```

6. Configure a VRF routing instance, and add IRB interface irb.1 to the routing instance.

```
[edit routing-instances]
user@router# set vrf instance-type vrf
user@router# set vrf interface irb.1
```

Verification

Verifying that a Logical Interface Has Correct ESI and EVPN Multihoming Mode

Purpose Verify that logical interface ae0.1 is configured with the correct ESI and EVPN multihoming mode.

Action From operational mode, enter the **show interfaces ae0.1** command.

```
user@switch>  show interfaces ae0.1
    Logical interface ae0.1 (Index 332) (SNMP ifIndex 706)
    Flags: Up SNMP-Traps 0x20004000 VLAN-Tag [ 0x8100.100 ]  Encapsulation:
VLAN-Bridge
    Input packets : 0
    Output packets: 0
    Protocol bridge, MTU: 1522
      Flags: Is-Primary
Ethernet segment value: 00:11:22:33:44:55:66:77:88:99, Mode: Single-active
```

Meaning The output shows that logical interface ae0.1 is configured with ESI 00:11:22:33:44:55:66:77:88:99 and the EVPN multihoming mode is single-active for multihoming active-standby mode. For a scenario with multihoming active-active mode, the output would display all-active.

Verifying the EVPN Routing Instance Status

Purpose Verify the status of the various elements configured in an EVPN routing instance.

Action From operational mode, enter the **show evpn instance extensive** command.

```
user@switch>  show evpn instance extensive
  Instance: blue
  ...
  Number of local interfaces:  (1 up)
    Interface name  ESI                             Mode             Status
AC-Role
    ae0.1           00:11:22:33:44:55:066:77:88:99  single-active       Up
   Root
  Number of IRB interfaces: 1 (1 up)
    Interface name  VLAN ID  Status  L3 context
    irb.1           1       Up      vrf
  Number of protect interfaces: 0
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label  SG sync  IM core nexthop
    1                 0    0                Local switching
   ...
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:066:77:88:99
      Status: Resolved by IFL ae0.1
      Local interface: ae0.1, Status: Up/Forwarding
      Number of remote PEs connected: 1
        Remote PE        MAC label  Aliasing label  Mode
        10.255.0.1       0          300928          all-active
      DF Election Algorithm: MOD based
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2
      Last designated forwarder update: Jul 28 13:04:29
      Advertised split horizon label: 300128
```

Meaning The output shows that for the EVPN routing instance named blue, the logical interface ae0.1 and IRB interface irb.1 are up and running. It also shows that VLAN 1 and Ethernet segment 00:11:22:33:44:55:066:77:88:99 are properly mapped to the routing instance. It also shows the status of the DF election.

Release History Table

Release	Description
15.1F6	Starting with Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface.

Related Documentation

- EVPN Multihoming Overview on page 29

CHAPTER 3

EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression

EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression

Proxy Address Resolution Protocol (ARP) and ARP suppression and Network Discovery Protocol (NDP) and NDP suppression are supported as follows:

- MX Series routers and EX9200 switches
  - Starting with Junos OS Release 17.2R1, MX Series routers and EX9200 switches that function as provider edge (PE) devices in an Ethernet VPN-MPLS (EVPN-MPLS) or Layer 3 VXLAN gateways in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment support proxy ARP and ARP suppression and NDP and NDP suppression on an integrated and routing (IRB) interface.
  - Starting with Junos OS Release 17.4R2, MX Series routers and EX9200 switches support proxy ARP and ARP suppression and NDP and NDP suppression on non-IRB interfaces. Junos OS Release 17.4R2 also introduces the ability to limit the number of media-access-control (MAC)-IP address bindings that can be learned on these Juniper Networks devices.
- QFX10000 switches—Starting with Junos OS Release 17.3R1, QFX10000 switches that function as Layer 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression and NDP and NDP suppression on an IRB interface.
- QFX5100, QFX5200, and QFX5110 switches—Starting with Junos OS Release 18.1R1, QFX5100 and QFX5200 switches that function as Layer 2 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression and NDP and NDP suppression on non-IRB interfaces. QFX5110 switches that function as Layer 2 or 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression and NDP and NDP suppression on IRB interfaces and non-IRB interfaces. You can also limit the number of MAC-IP address bindings that can be learned on these switches.

This feature reduces the flooding of ARP and NDP messages in the EVPN network, resulting in a more efficient use of core bandwidth. By default, proxy ARP and ARP suppression and NDP and NDP suppression are enabled.

When limiting the number of MAC-IP address bindings that can be learned, the limit can be configured globally or for a specific routing instance, bridge domain, VLAN, or interface. After the specified limit is reached, no additional entries are added to the MAC-IP binding database. You an also specify a timeout interval for MAC-IP address bindings.

Proxy ARP and NDP snooping are enabled by default for all EVPN-MPLS or EVPN-VXLAN bridge domains and VLANs. ARP or NDP packets generated from a local customer edge (CE) device or Layer 2 VXLAN gateway are snooped. ARP and NDP packets generated from a remote PE device or Layer 3 VXLAN gateway through core-facing interfaces, however, are not snooped.

Both IRB and non-IRB interfaces configured on a PE device or a Layer 2 or 3 VXLAN gateway deliver ARP requests and NDP requests. When one of these devices receives an ARP request or NDP request, the device searches its MAC-IP address bindings database for the requested IP address. If the device finds the MAC-IP address binding in its database, it responds to the request. If the device does not find the MAC-IP address binding, it takes the following action:

- If the device is running Junos OS Releases 17.2R*x* or 17.3R*x*, the device swaps the source MAC address with the MAC address of the interface on which the request was received and sends the request to all interfaces.
- If the device is running Junos OS Releases 17.4R1 or later, the device leaves the source MAC address as is and sends the request to all interfaces.

Even when a PE device or a Layer 2 or 3 VXLAN gateway responds to an ARP request or NDP request, ARP packets and NDP might still be flooded across the WAN. ARP suppression and NDP suppression prevent this flooding from occurring.

You can disable the suppression of ARP packets and NDP packets by specifying the **no-arp-suppression** configuration statement. However, if you do so, be aware of the following implications:

- ARP and NDP packets will be flooded.
- The PE device or Layer 2 or 3 VXLAN gateway does not respond to ARP or NDP requests.
- Then PE device or Layer 2 or 3 VXLAN gateway does not learn the IP address from the ARP or NDP request.

Therefore, we recommend that ARP suppression and NDP suppression remain enabled.

Proxy ARP and ARP suppression and NDP and NDP suppression are supported in the following scenarios:

- Single-homed devices in active mode—EVPN-MPLS and EVPN-VXLAN
- Multihomed devices in active-active mode—EVPN-MPLS and EVPN-VXLAN
- Multihomed devices in single-active mode—EVPN-MPLS only

In a multihoming active-active scenario, the database of MAC-IP address bindings are synchronized between the PE device or Layer 2 or 3 VXLAN gateway that act as the designated forwarder (DF) and non-designated forwarder (non-DF).

Release History Table

Release	Description
18.1R1	Starting with Junos OS Release 18.1R1, QFX5100 and QFX5200 switches that function as Layer 2 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression and NDP and NDP suppression on non-IRB interfaces. QFX5110 switches that function as Layer 2 or 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression and NDP and NDP suppression on IRB interfaces and non-IRB interfaces. You can also limit the number of MAC-IP address bindings that can be learned on these switches.
17.4R2	Starting with Junos OS Release 17.4R2, MX Series routers and EX9200 switches support proxy ARP and ARP suppression and NDP and NDP suppression on non-IRB interfaces.
17.4R2	Junos OS Release 17.4R2 also introduces the ability to limit the number of media-access-control (MAC)-IP address bindings that can be learned on these Juniper Networks devices.
17.3R1	Starting with Junos OS Release 17.3R1, QFX10000 switches that function as Layer 3 VXLAN gateways in an EVPN-VXLAN environment support proxy ARP and ARP suppression and NDP and NDP suppression on an IRB interface.
17.2R1	Starting with Junos OS Release 17.2R1, MX Series routers and EX9200 switches that function as provider edge (PE) devices in an Ethernet VPN-MPLS (EVPN-MPLS) or Layer 3 VXLAN gateways in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment support proxy ARP and ARP suppression and NDP and NDP suppression on an integrated and routing (IRB) interface.

Related Documentation

- global-mac-ip-limit on page 1059
- interface-mac-ip-limit on page 1065
- mac-ip-table-size on page 1074
- no-arp-suppression on page 1078

CHAPTER 4

Configuring MAC Pinning

EVPN MAC Pinning Overview

Starting in Release 16.2, Junos OS enables MAC address pinning for Ethernet VPN (EVPN), including customer edge (CE) interfaces and EVPN over MPLS core interfaces, in both all-active mode or active-standby mode.

When you configure an interface with MAC pinning, the l2ald process adds an 8-octet extension to the address (which implements aspects of Section 7.7 of RFC-7432: MAC Mobility Extended Community). The low-order bit in the flag octet of this structure is the Sticky/static flag. Configuring MAC pinning sets the flag to 1 and designates the address is static.

If you configure MAC pinning on a CE/VPLS interface, that MAC address cannot be moved to any other CE/VPLS interface. Similarly, if you configure MAC pinning on an MPLS core interface on a PE device, that MAC address cannot be moved to a different interface on the MPLS core.

CE devices advertise MAC pinned addresses through the l2ald process to remote PE devices. When a PE device learns a MAC-pinned interface address from a CE device, the PE device synchronizes the address, through the control plane, with remote peer PE devices in the EVPN network. Thereafter, if the PE device receives traffic, or an advertised route, from any CE device in the EVPN network that bears the same source MAC address, the receiving device drops the traffic.

A PE device that learns a MAC pinned address via its control plane prefers that address over an address bearing the identical MAC address that is learned from a remote peer PE device, or locally by way of its l2ald from a CE/VPLS interface.

Before the introduction of MAC address pinning for EVPN, a MAC address on a remote PE device that was learned locally could be aged out. For EVPN MAC pinning, a pinned MAC address does not age out on the remote PE device unless the routing protocol process removes it from the routing table. Similarly, a pinned MAC address persists for the control plane of the remote PE device.

This static interface address cannot be moved unless it is deleted from the routing table of the device on which it is configured.

A MAC address might be considered moved as a result of:

- Misconfiguration
- Configuration on a different Ethernet segment
- Physical movement of a device within a network topology

NOTE: If EVPN is configured in all-active multihoming mode, you must either enable or disable MAC pinning on the multihoming PE device interfaces in the broadcast domain. Also, either enable or disable MAC pinning on all CE device interfaces to avoid inconsistent MAC learning in the EVPN broadcast domain.

If EVPN is configured in active-standby multihoming mode, a MAC pinned address received by the active PE device can be moved to a CE interface on the standby PE device in response to a switchover, if traffic has been running continuously on the CE interface.

CAUTION: Do not enable or disable MAC pinning on an interface while traffic is running.

Release History Table

Release	Description
16.2	Starting in Release 16.2, Junos OS enables MAC address pinning for Ethernet VPN (EVPN), including customer edge (CE) interfaces and EVPN over MPLS core interfaces, in both all-active mode or active-standby mode.

Related Documentation

- EVPN Overview on page 549
- Configuring EVPN Routing Instances on page 3
- *Understanding MAC Pinning*
- mac-pinning on page 1075

Configuring EVPN MAC Pinning

The EVPN MAC pinning feature enables you to apply MAC pinning to CE/VPLS interfaces in a bridge domain and to PE interfaces in the core of a service-provider network.

The EVPN MAC pinning feature enables you to make the MAC address associated with a specific interface static. That is, the MAC-pinned address cannot be moved to any other interface in the bridge domain. The MAC-pinned address on the interface will not age out.

Traffic transmitted with pinned MAC address as source MAC, from any interface in the EVPN domain other than the one on which the MAC address is pinned, will be discarded.

Pinning a MAC address to an interface for EVPN does not require new CLI. Existing CLI enables pinning MAC addresses to CE-device and PE-device interfaces. You can enable MAC Pinning on and interface in an EVPN network with the following command:

User@CE1# set switch-options interface *interface-name* mac-pinning

Related Documentation

- EVPN Overview on page 549
- Configuring EVPN Routing Instances on page 3
- EVPN MAC Pinning Overview on page 232
- **mac-pinning on page 1075**

CHAPTER 5

NSR and Unified ISSU

NSR and Unified ISSU Support for EVPN Overview

Starting in Release 16.2, Junos OS ensures minimal loss of traffic when a Routing Engine switchover occurs with nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) enabled. The forwarding state of the Packet Forwarding Engine (PFE) remains intact during switchover. The signaling state on the primary Routing Engine and on the standby Routing Engine are built in parallel.

EVPN reproduces dynamically generated data (such as labels and sequence numbers), and data obtained from peers on the primary Routing Engine, and on the standby Routing Engine. EVPN also monitors BGP ingress and egress routing table messages on the standby Routing Engine to populate its signaling plane data structures. Local MAC addresses are obtained by the Layer 2 address learning process (l2ald), which transfers the data to the EVPN module in the route processing software. In the network layer reachability information (NLRI) fields of its packets, BGP transfers the MAC addresses to peers in the network.

In earlier releases, the l2ald did not run on the standby Routing Engine. Consequently, the l2ald did not inform the routing protocol process on the standby Routing Engine about locally learned MAC addresses. With this feature, the routing protocol process learns acquires-learned MAC addresses by listening to the BGP ingress and egress routing table messages and then populates its data structures.

Following a switchover, when the l2ald becomes active on the new primary Routing Engine, it sends all locally-learned MAC addresses to the routing protocol process, which can then verify the MAC addresses learned from the l2ald with MAC addresses derived from the BGP routing table egress messages.

NOTE: **Expect a traffic loss pertaining to a topology change if the topology change occurs during a switchover.**

This feature mirrors the following data to the standby Routing Engine:

- MAC route labels—EVPN allocates a MAC route label per EVPN instance (EVI) and per Ethernet segment Identifier (ESI). MAC labels, including the EVI and ESI are mirrored to the standby Routing Engine.
- Inclusive multicast (IM) route labels—EVPN allocates an IM label per VLAN. An IM label, including the EVI name and VLAN ID are mirrored to the standby Routing Engine.
- Split horizon labels—EVPN mirrors a split horizon label and the EVI name to the standby Routing Engine.
- Aliasing labels—EVPN mirrors an aliasing label and EVI name to the standby Routing Engine.
- Dummy labels—EVPN creates a dummy label with which it adds the egress route to the mpls.0 table. EVPN mirrors a dummy label and the EVI name to the standby Routing Engine so the mpls.0 table is identical on the primary and standby Routing Engines.

For EVPN ETREE, Junos mirrors the local EVPN ETREE leaf label that is advertised to other PE as part of the ETREE extended community on the standby Routing Engine.

For EVPN P2MP, Junos mirrors the LDP/RSVP transport label on the standby Routing Engine.

NSR Data Flow Pre-Switchover

The EVPN configuration on the standby Routing Engine directs it to operate identically to the primary Routing Engine except that it does not allocate any labels. Label information is updated on the standby Routing Engine when it receives the information from its label information base mirror (libmirror).

BGP updates remote MAC addresses in the instance.EVPN table on the standby Routing Engine. Just as EVPN operates on the primary Routing Engine, the standby Routing Engine derives information from its flash drive to update its data structures.

To gather local MAC addresses and update data structures, EVPN on the standby Routing Engine listens to routing information base (RIB, also known as routing table) egress messages. All MAC addresses that the standby Routing Engine learns from the BGP RIB egress messages are marked as stale. Following a switchover, when the l2ald becomes active, if it sends MAC addresses that are identical to those marked stale, the addresses are retained. If the l2ald does not send the same MAC addresses learned from BGP RIB egress messages, it deletes the addresses.

NSR Data Flow Post Switchover

Following a switchover, when the standby Routing Engine becomes the primary Routing Engine, it establishes connection with the l2ald. When the l2ald becomes active, it sends local MAC addresses to the routing protocol process. The routing protocol process removes the stale bit from the MAC addresses it receives from the l2ald.

MAC addresses that were not derived from BGP RIB egress messages, but were sent to the routing protocol process by the l2ald, are treated as newly learned MAC addresses. BGP advertises such MAC addresses.

When the l2ald sends the end marker of the MAC address to the routing protocol process, all local MAC addresses marked as stale are deleted.

Unified ISSU Support

Starting in Junos OS Release 17.2R1, unified in-service software upgrade is supported for VXLAN on MX Series routers. ISSU enables you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Unified ISSU is supported only on dual Routing Engine platforms. In addition, the graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) features must be enabled. Unified ISSU allows you to eliminate network downtime, reduce operating costs, and deliver higher levels of services. See *Getting Started with Unified In-Service Software Upgrade*.

To enable GRES, include the **graceful-switchover** statement at the **[edit chassis redundancy]** hierarchy level.

To enable NSR, include the **nonstop-routing** statement at the [**edit routing-options**] hierarchy level and the **commit synchronize** statement at the **[edit system]** hierarchy level.

Supported Features on EVPN

Junos OS supports NSR/ISSU on the following features for EVPN:

Feature	Initial Junos OS Release
EVPN with ingress replication for BUM traffic	16.2R1
EVPN-ETREE	17.2R1
EVPN-VPWS	17.2R1
EVPN -VXLAN	17.2R1
PBB-EVPN	17.2R1
EVPN with P2MP mLDP replication for BUM traffic	18.2R1

Release History Table

Release	Description
17.2R1	Starting in Junos OS Release 17.2R1, unified in-service software upgrade is supported for VXLAN on MX Series routers.
16.2	Starting in Release 16.2, Junos OS ensures minimal loss of traffic when a Routing Engine switchover occurs with nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) enabled.

Related Documentation

- show evpn instance on page 1163
- show evpn database on page 1157

- Tracing EVPN Traffic and Operations on page 19
- *Understanding Graceful Routing Engine Switchover*

Preventing Traffic Loss in an EVPN/VXLAN Environment With GRES and NSR

The graceful Routing Engine switchover (GRES) feature in Junos OS enables QFX10000 switches to continue forwarding packets, even if one Routing Engine fails. To preserve routing during a switchover, GRES is combined with nonstop active routing (NSR). Although GRES preserves interface and kernel information, in an EVPN/VXLAN environment, a QFX10000 switch could experience Layer 3 traffic loss if traffic is flowing during the switchover or reboot of the Routing Engine.

To prevent traffic flows from being dropped in such a scenario, configure a hold-time interval that prevents IP phantom addresses from being added to the routing tables. During this interval, the routes are maintained in the kernel. After the interval expires, the phantom routes are added to the appropriate routing tables before they are deleted from the kernel.

NOTE: **The recommended hold-time value in an EVPN/VXLAN environment is 300 seconds (5 minutes).**

- Enable GRES
- Enable NSR

To specify a hold-time interval that prevents phantom routes from being added to the routing table during a switchover until the interval expires:

1. Specify a hold-time interval of 300.

```
[edit routing-options]
user@swtich# set nsr-phantom-holdtime seconds 300.
```

NOTE: **The hold-time interval range is 0 through 10000.**

Related Documentation

- *Understanding Graceful Routing Engine Switchover*

CHAPTER 6

Monitoring EVPN Networks

Connectivity Fault Management Support for EVPN and Layer 2 VPN Overview

The IEEE 802.1ag specification provides for Ethernet connectivity fault management (CFM). The goal of CFM is to monitor an Ethernet network that consists of one or more service instances through the use of CFM protocol messages. CFM partitions the service network into various administrative domains. Each administrative domain is mapped into a maintenance domain. A maintenance association end point (MEP) refers to the boundary of a domain. A MEP generates and responds to CFM protocol messages. You can configure multiple up (PE to PE) MEPs or down (PE to CE) MEPs for a single instance of a maintenance domain identifier and a maintenance association name to monitor services in a VPN.

For Layer 2 VPNs and EVPN networks, you can configure multiple up MEPs for a single combination of maintenance association ID and maintenance domain ID on a routing instance on the logical interfaces (IFLs), regardless of whether the logical interface is composed of physical interfaces on the same device or on different devices. The devices must be in enhanced IP network services mode.

In an EVPN network, the following CFM features are supported:

- Monitoring the connectivity between two provider edge (PE) routers in an active-active or active-standby multihomed configuration.
- Delay measurement and Synthetic Loss measurement. This feature is not supported when multiple MEPs are configured on multiple logical interfaces (IFLs) on the same physical interface (IFD).
- CFM monitoring between PE devices and customer edge (CE) devices. When the customer edge device is not a Juniper Networks device, you can enable CFM monitoring by using either the remote defect indication (RDI) bit or the Interface Status TLV. For more information, see *Understanding CFM Monitoring between CE and PE Devices*

LImitations of CFM on layer 2 VPN and EVPNs

- In a circuit cross-connect (ccc) layer 2 VPN or local switch with MEPs and maintenance intermediate points (MIPs), the counter for link trace messages (LTMs) received on the MAC address of the up MEP does not get incremented when the MIP in the path is configured at the same level. The MIP traps the LTM packet, while the LTR message is sent. This leads to a discrepancy between the number of LTMs received and the number of LTRs sent.
- CFM up MEP on an EVPN network does not support the use of action profiles for interface down. In other words, you can configure an action profile, but no action is taken.
- CFM up MEP is only supported on EVPN E-LAN services. EVPN-ELINE and EVPN-TREE services do not support up MEP CFM.

For more information on CFM, see IEEE 802.1ag OAM Connectivity Fault Management Overview.

Related Documentation

- *IEEE 802.1ag OAM Connectivity Fault Management Overview*
- *Creating a Maintenance Domain*
- *Creating a Maintenance Association*
- Configuring a MEP to Generate and Respond to CFM Protocol Messages on page 240
- *show oam ethernet connectivity-fault-management interfaces*

Configuring a MEP to Generate and Respond to CFM Protocol Messages

A maintenance association end point (MEP) refers to the boundary of a domain. A MEP generates and responds to connectivity fault management (CFM) protocol messages. You can configure multiple up MEPs for a single combination of maintenance association ID and maintenance domain ID for interfaces belonging to a particular VPLS service or a bridge domain. You can configure multiple down MEPs for a single instance of maintenance domain identifier and maintenance association name to monitor services provided by Virtual Private LAN service (VPLS), bridge domain, circuit cross-connect (CCC), or IPv4 domains.

For layer 2 VPNs routing instances (local switching) and EVPN routing instances, you can also configure multiple up MEPs for a single combination of maintenance association ID and maintenance domain ID on logical interfaces.. The logical interface can be configured on different devices or on the same device. To support multiple up MEPs on two IFLs, enhanced IP network services must be configured for the chassis.

You can enable automatic discovery of a MEP. With automatic discovery a MEP is enabled to accept continuity check messages (CCMs) from all remote MEPs of the same maintenance association. if automatic discovery is not enabled, the remote MEPs must be configured. If the remote MEP is not configured, the CCMs from the remote MEP are treated as errors.

Continuity measurement is provided by an existing continuity check protocol. The continuity for every remote MEP is measured as the percentage of time that remote MEP was operationally up over the total administratively enabled time. Here, the operational uptime is the total time during which the CCM adjacency is active for a particular remote MEP and the administrative enabled time is the total time during which the local MEP is active. You can also restart the continuity measurement by clearing the currently measured operational uptime and the administrative enabled time.

- Configuring a Maintenance Association End Point (MEP) on page 241
- Configuring a remote Maintenance Association End Point (MEP) on page 243

Configuring a Maintenance Association End Point (MEP)

To configure a maintenance association end point:

1. Specify an ID for the MEP at the [**edit protocols oam ethernet connectivity-fault-management maintenance-domain** *domain-name* **maintenance-association** *ma-name*]. You can specify any value from 1 through 8191.

   ```
   [edit protocols oam ethernet connectivity-fault-management maintenance-domain
     domain-name maintenance-association ma-name]
   user@host# set mep mep-id
   ```

2. Enable maintenance end point automatic discovery so the MEP can accept continuity check messages (CCMs) from all remote MEPs of the same maintenance association.

   ```
   [edit protocols oam ethernet connectivity-fault-management maintenance-domain
     domain-name maintenance-association ma-name mep mep-id]
   user@host# set auto-discovery
   ```

3. Specify the direction in which the CCM packets are transmitted for the MEP. You can specify up or down. If you specify the direction as up, CCMs are transmitted out of every logical interface that is part of the same bridging or VPLS instance except for the interface configured on the MEP. If you specify the direction as down, CCMs are transmitted only out of the interface configured on the MEP.

   NOTE: Ports in the Spanning Tree Protocol (STP) blocking state do not block CFM packets destined to a down MEP. Ports in an STP blocking state without the continuity check protocol configured do block CFM packets.

   ```
   [edit protocols oam ethernet connectivity-fault-management maintenance-domain
     domain-name maintenance-association ma-name mep mep-id]
   user@host# set direction down
   ```

NOTE: Starting with Junos OS Release 12.3, for all interfaces configured on Modular Port Concentrators (MPCs) on MX Series 5G Universal Routing Platforms, you no longer need to configure the **no-control-word** statement for all Layer 2 VPNs and Layer 2 circuits over which you are running CFM MEPs. For all other interfaces on MX Series routers and on all other routers and switches, you must continue to configure the **no-control-word** statement at the **[edit routing-instances** *routing-instance-name* **protocols l2vpn]** or **[edit protocols l2circuit neighbor** *neighbor-id* **interface** *interface-name***]** hierarchy level when you configure CFM MEPs. Otherwise, the CFM packets are not transmitted, and the **show oam ethernet connectivity-fault-management mep-database** command does not display any remote MEPs.

4. Specify the interface to which the MEP is attached. It can be a physical interface, logical interface, or trunk interface. On MX Series routers, the MEP can be attached to a specific VLAN of a trunk interface.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain
  domain-name maintenance-association ma-name mep mep-id]
user@host# set interface interface-name
```

5. Specify the IEEE 802.1 priority bits that are used by continuity check and link trace messages. You can specify a value from through 7 as the priority.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain
  domain-name maintenance-association ma-name mep mep-id]
user@host# set priority number
```

6. Specify the lowest priority defect that generates a fault alarm whenever CFM detects a defect. Possible values include: all -defects, err-xcon, mac-rem-err-xcon, no-defect, rem-err-xcon, and xcon.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain
  domain-name maintenance-association ma-name mep mep-id]
user@host# set lowest-priority-defect mac-rem-err-xcon
```

7. Specify the ID of the remote MEP at the **[edit protocols oam ethernet connectivity-fault-management maintenance-domain** *domain-name* **maintenance-association** *ma-name* **mep** *mep-id***]**. You can specify any value from 1 through 8191.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain
  domain-name maintenance-association ma-namemep mep-id]
user@host# set remote-mep mep-id
```

See Also

- *auto-discovery*
- *direction*
- *lowest-priority-defect*
- *priority*

Configuring a remote Maintenance Association End Point (MEP)

To configure a remote maintenance association end point:

1. Configure the remote MEP by specifying the MEP ID at the [**edit protocols oam ethernet connectivity-fault-management maintenance-domain** *domain-name* **maintenance-association** *ma-name* **mep** *mep-id***]**. You can specify any value from 1 through 8191.

   ```
   [edit protocols oam ethernet connectivity-fault-management maintenance-domain
     domain-name maintenance-association ma-namemep mep-id]
   user@host# edit remote-mep mep-id
   ```

2. Specify the name of the action profile to be used for the remote MEP by including the **action-profile** *profile-name* statement at the [**edit protocols oam ethernet connectivity-fault-management maintenance-domain** *domain-name* **maintenance-association** *ma-name* **mep** *mep-id* **remote-mep** *remote-mep-id***]**. The profile must be defined at the [**edit protocols oam ethernet connectivity-fault-management**] hierarchy level.

   ```
   [edit protocols oam ethernet connectivity-fault-management maintenance-domain
     domain-name maintenance-association ma-namemep mep-id remote-mep
     remote-mep-id]
   user@host# set action-profile profile-name
   ```

3. Configure the remote MEP to detect initial loss of connectivity. By default, the MEP does not generate loss-of-continuity (LOC) defect messages. When you configure the **detect-loc** statement, a loss-of-continuity (LOC) defect is detected if no continuity check message is received from the remote MEP within a period equal to 3.5 times the continuity check interval configured for the maintenance association. If a LOC defect is detected, a syslog error message is generated.

   NOTE: When you configure connectivity-fault management (CFM) along with detect-loc, any action-profile configured to bring down the interface is executed if continuity check message is not received . However, the action-profile is not executed if you have not configured detect-loc and continuity check message is not received.

```
[edit protocols oam ethernet connectivity-fault-management maintenance-domain
   domain-name maintenance-association ma-namemep mep-id remote-mep
   remote-mep-id]
user@host# set detect-loc
```

See Also

- *action-profile*
- *detect-loc*
- *remote-mep*

Release History Table

Release	Description
12.3	Starting with Junos OS Release 12.3, for all interfaces configured on Modular Port Concentrators (MPCs) on MX Series 5G Universal Routing Platforms, you no longer need to configure the **no-control-word** statement for all Layer 2 VPNs and Layer 2 circuits over which you are running CFM MEPs.

Related Documentation

- *connectivity-fault-management*
- *IEEE 802.1ag OAM Connectivity Fault Management Overview*
- *Configuring a CFM Action Profile to Specify CFM Actions for CFM Events*
- *Configuring Port Status TLV and Interface Status TLV*
- *Configuring MAC Flush Message Processing in CET Mode*
- *Configuring M120 and MX Series Routers for CCC Encapsulated Packets*
- *Configuring Rate Limiting of Ethernet OAM Messages*
- *Configuring Service Protection for VPWS over MPLS Using the MEP Interface*

PART 2

EVPN-VXLAN

CHAPTER 7

Overview

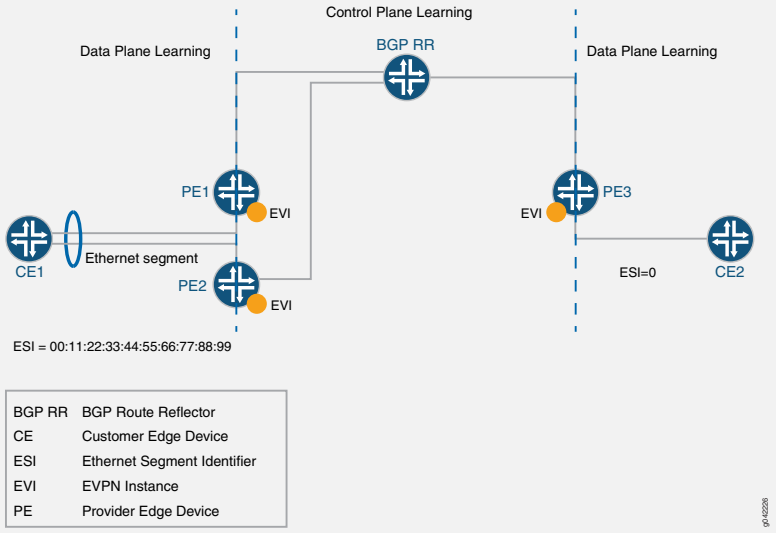
Understanding EVPN with VXLAN Data Plane Encapsulation

Ethernet VPNs (EVPNs) enable you to connect groups of dispersed customer sites using Layer 2 virtual bridges, and Virtual Extensible LANs (VXLANs) allow you to stretch Layer 2 connectivity over an intervening Layer 3 network, while providing network segmentation like a VLAN, but without the scaling limitation of traditional VLANs. EVPN with VXLAN encapsulation handles Layer 2 connectivity at the scale required by cloud server providers and replaces limiting protocols like Spanning Tree Protocol (STP), freeing up your Layer 3 network to use more robust routing protocols. EVPN with VXLAN data plane encapsulation can be used with and without Juniper Networks Contrail virtualization software—use Contrail with EVPN VXLAN data plane encapsulation when you have an environment that includes both virtual and bare-metal devices.

Understanding EVPN

Ethernet VPN (EVPN) is a standards-based technology that provides virtual multipoint bridged connectivity between different Layer 2 domains over an IP or IP/MPLS backbone network. Like other VPN technologies, such as IP VPN and virtual private LAN service (VPLS), EVPN instances are configured on provider edge (PE) routers to maintain logical service separation between customers. The PE routers connect to customer edge (CE) devices, which can be routers, switches, or hosts. The PE routers then exchange reachability information using Multiprotocol BGP (MP-BGP) and encapsulated traffic is forwarded between PE routers. Because elements of the architecture are common with other VPN technologies, you can seamlessly introduce and integrate EVPN into existing service environments, as shown in Figure 15 on page 248.

Figure 15: EVPN Overview



The EVPN is used as a Layer 2 overlay solution to provide Layer 2 connection over an IP underlay for the endpoints within a virtual network whenever Layer 2 connectivity is required by an end station such as bare-metal server (BMS). Otherwise, Layer 3 routing is used through VRF tables between Contrail vRouters and MX Series routers. EVPN technology offers multitenancy, flexible services that can be extended on demand, frequently using compute resources of different physical data centers for a single service (Layer 2 extension).

EVPN's MP-BGP control plane enables you to dynamically move live virtual machines from one data center to another, also known as virtual machine (VM) motion. After you move a VM to a destination server or hypervisor, it transmits a gratuitous ARP, which updates the Layer 2 forwarding table of the PE device at the destination data center. The PE device then transmits a MAC route update to all remote PE devices which in turn update their forwarding tables. An EVPN tracks the movement of the VM, which is also known as MAC mobility.

EVPN also has mechanisms that detect and stop MAC flapping, and prevent the looping of broadcast, unknown unicast, and multicast (BUM) traffic in an all-active multi-homed topology.

The EVPN technology, similar to Layer 3 MPLS VPN, includes the concept of routing MAC addresses using IP/MPLS core. EVPN provides the following benefits:

- Ability to have an active multihomed edge device
- Aliasing
- Fast convergence
- Load balancing across dual-active links
- MAC address mobility
- Multitenancy

In addition, EVPN uses these techniques:

- *Multihoming* provides redundancy in the event that an access link or one of the PE routing devices fails. In either case, traffic flows from the CE device towards the PE router, using the remaining active links. For traffic in the other direction, the remote PE router updates its forwarding table to send traffic to the remaining active PE routers connected to the multihomed Ethernet segment. EVPN provides a fast convergence mechanism, which reduces traffic restoration time so that the time it takes to make this adjustment is independent of the number of media access control (MAC) addresses learned by the PE router. All-active multihoming enables a CE device to connect to two or more PE routers such that traffic is forwarded using all of the links between the devices. This multihoming enables the CE device to load-balance traffic to multiple PE routers. More importantly, multihoming enables a remote PE router to load-balance traffic to the multihomed PE routers across the core network. This load balancing of traffic flows between data centers is known as *aliasing*, which causes different signals to become indistinguishable—they become aliases of one another. Aliasing is used with digital audio and digital images.
- *Split horizon* prevents the looping of broadcast, unknown unicast, and multicast (BUM) traffic in a network. The split horizon basic principle is simple: Information about the routing for a particular packet is never sent back in the direction from which it was received.
- *Local link bias* conserves bandwidth by using local links to forward unicast traffic exiting a Virtual Chassis or Virtual Chassis Fabric (VCF) that has a link aggregation group (LAG) bundle composed of member links on different member switches in the same Virtual Chassis or VCF. A local link is a member link in the LAG bundle that is on the member switch that received the traffic.
- EVPN with VXLAN encapsulation is used for Layer 2 connectivity between virtual machines and a top-of-rack (TOR) switch, for example, a QFX5100 switch, within a Layer 2 domain.

You can use Contrail to provision an MX Series router as a Layer 2 or Layer 3 VXLAN gateway. MX Series routers implement the NETCONF XML management protocol, which

is an XML-based protocol that client applications use to request and change configuration information on routing, switching, and security devices. The NETCONF XML management protocol uses an XML based data encoding for the configuration data and remote procedure calls. The NETCONF protocol defines basic operations that are equivalent to configuration mode commands in the command-line interface (CLI). Applications use the protocol operations to display, edit, and commit configuration statements similarly to how administrators use CLI configuration mode commands to perform those same operations.

Understanding VXLAN

Virtual extensible LANs (VXLANs) introduced an overlay scheme that expands the Layer 2 network address space from 4K to 16 million, largely solving the scaling issues seen in VLAN-based environments.

Network overlays are created by encapsulating traffic and tunneling the traffic over a physical network. You can use a number of tunneling protocols in the data center to create network overlays—the most common protocol is VXLAN. VXLAN tunneling protocol encapsulates Layer 2 Ethernet frames in Layer 3 UDP packets. This encapsulation enables you to create virtual Layer 2 subnets or segments that can span physical Layer 3 networks.

In a VXLAN overlay network, a VXLAN network identifier (VNI) uniquely identifies each Layer 2 subnet or segment. A VNI segments traffic the same way that an IEEE 802.1Q VLAN ID segments traffic. As is the case with VLAN, virtual machines on the same VNI can communicate directly with each other, whereas virtual machines on different VNIs need a router to communicate with each other.

The entity that performs the encapsulation and de-encapsulation is called a VXLAN tunnel endpoint (VTEP). In the physical network, a Juniper Networks device that functions as a Layer 2 or Layer 3 VXLAN gateway can enacapsulate and de-encapsulate data packets. This type of VTEP is known as a hardware VTEP. In the virtual network, VTEPs can reside in hypervisor hosts, such as kernel-based virtual machine (KVM) hosts. This type of VTEP is known as a software VTEP.

Each VTEP has two interfaces.

- One interface is a switching interface that faces the virtual machines in the host and provides communication between VMs on the local LAN segment.
- The other is an IP interface that faces the Layer 3 network.

Each VTEP has a unique IP address that is used for routing the UDP packets between VTEPs. For example, when VTEP1 receives an Ethernet frame from VM1 addressed to VM3, it uses the VNI and the destination MAC to look up in its forwarding table which VTEP sends the packet to. It then adds a VXLAN header that contains the VNI to the Ethernet frame and encapsulates the frame in a Layer 3 UDP packet and routes the packet to VTEP2 over the Layer 3 network. VTEP2 de-encapsulates the original Ethernet frame and forwards it to VM3. VM1 and VM3 cannot detect the VXLAN tunnel and the Layer 3 network between them.

EVPN-VXLAN Integration Overview

VXLAN defines a tunneling scheme to overlay Layer 2 networks on top of Layer 3 networks. This tunneling scheme allows for optimal forwarding of Ethernet frames with support for multipathing of unicast and multicast traffic with the use of UDP/IP encapsulation for tunneling, and is mainly used for the intra-data center site connectivity.

A unique characteristic of EVPN is that MAC address learning between PE routers occurs in the control plane. The local PE router detects a new MAC address from a CE device and then, using MP-BGP, advertises the address to all the remote PE routers. This method differs from existing Layer 2 VPN solutions such as VPLS, which learn by flooding unknown unicast in the data plane. This control plane MAC learning method is the key enabler of the many useful features that EVPN provides.

Because MAC learning is handled in the control plane, EVPN has the flexibility to support different data plane encapsulation technologies between PE routers. This flexibility is important because not every backbone network might be running MPLS, especially in enterprise networks.

EVPN addresses many of the challenges faced by network operators building data centers to offer cloud and virtualization services. The main application of EVPN is Data Center Interconnect (DCI), which refers to the ability to extend Layer 2 connectivity between different data centers that are deployed to improve the performance of delivering application traffic to end users and for disaster recovery.

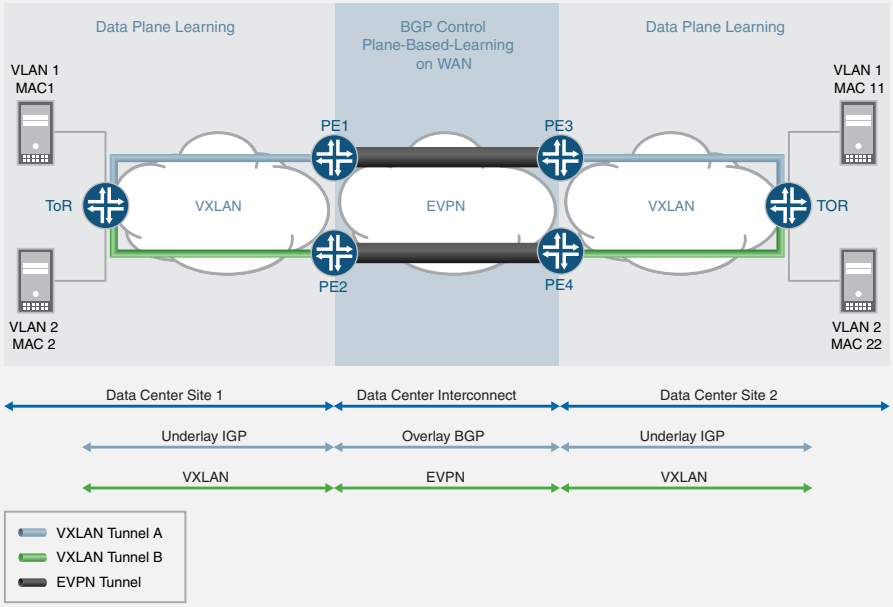
Although various DCI technologies are available, EVPN has an advantage over the other MPLS technologies because of its unique features, such as active/active redundancy, aliasing, and mass MAC withdrawal. As a result, to provide a solution for DCI, VXLAN is integrated with EVPN.

As shown in "Overview" on page 247, each VXLAN, which is connected to the MPLS or IP core, runs an independent instance of the interior gateway protocol (IGP) control plane. Each PE router participates in the IGP control plane instance of its VXLAN. Each customer is a data center, so each has its own virtual router for VXLAN underlay.

Each PE node can terminate the VXLAN data plane encapsulation where the VXLAN network identifier (VNI) is mapped to a bridge domain or VLAN. The PE router performs data plane learning on the traffic received from the VXLAN.

Each PE node implements EVPN to distribute the client MAC addresses learned over the VXLAN tunnel into BGP. Each PE node encapsulates the VXLAN or Ethernet frames with MPLS when sending the packets over the MPLS core and with the VXLAN tunnel header when sending the packets over the VXLAN network.

Figure 16: EVPN-VXLAN Integration Overview



Firewall Filtering and Policing Support for EVPN-VXLAN

(QFX5100, QFX5100 Virtual Chassis, and QFX5110 switches) Starting with Junos OS Release 18.2R1, you can configure firewall filters and policers on VXLAN traffic in an EVPN topology. For each firewall filter that you apply to a VXLAN, you can specify **family ethernet-switching** to filter Layer 2 (Ethernet) packets, or **family inet** to filter on IRB interfaces. The IRB interface acts as a Layer 3 routing interface to connect the VXLANs in one-layer or two-layer IP fabric topologies. For more information, see *Configuring Firewall Filters*.

(QFX10000 switches) Starting with Junos OS Release 18.3R1, you can configure a firewall filter on an IRB interface in an EVPN-VXLAN topology. Firewall filters can only be configured on the IRB interface after the VXLAN header is stripped by the VXLAN tunnel endpoint (VTEP). For a list of supported match conditions, see *Firewall Filter Match Conditions and Actions for QFX10000 Switches*.

Understanding Contrail Virtual Networks Use with EVPN-VXLAN

Juniper Networks Contrail virtualization software is a software-defined networking (SDN) solution that automates and orchestrates the creation of highly scalable virtual networks. These virtual networks enable you to harness the power of the cloud—for new services, increased business agility, and revenue growth. MX Series routers can use EVPN-VXLAN to provide both Layer 2 and Layer 3 connectivity for end stations within a Contrail virtual network (VN).

NOTE: Contrail does not support the provisioning of Layer 2 and Layer 3 gateways on QFX Series switches.

The Contrail software for virtual networks provides both Layer 2 and Layer 3 connectivity. With Contrail, Layer 3 routing is preferred over Layer 2 bridging whenever possible. Layer 3 routing is used through virtual routing and forwarding (VRF) tables between Contrail vRouters and physical MX Series routers. MX Series routers provide Layer 3 gateway functionality between virtual networks.

Contrail enables you to use EVPN-VXLAN when your network includes both virtual and bare-metal devices.

Two types of encapsulation methods are used in virtual networks.

- MPLS-over-GRE (generic routing encapsulation) is used for Layer 3 routing between Contrail and MX Series routers.
- EVPN-VXLAN is used for Layer 2 connectivity between virtual machines and top-of-rack (TOR) switches, for example, QFX5100 switches, within a Layer 2 domain. For Layer 2 connectivity, traffic load balancing in the core is achieved by using the multihoming all-active feature provided by EVPN. Starting with Junos OS Release 17.3R1, EX9200 switches also support EVPN-VXLAN with Contrail.

NOTE: MPLS core is not supported on switches—only MX Series routers support this feature.

You cannot simultaneously mix EVPN-VXLAN with Open vSwitch Database (OVSDB)-VXLAN on QFX Series switches. After a switch is set to OVSDB-managed, the controller treats all ports as managed by OVSDB.

EVPN-VXLAN Support for VXLAN Underlay

Starting in Junos OS Release 14.1, MX Series routers support Virtual Extensible LAN (VXLAN) gateways. Each VXLAN gateway supports the following functionalities:

- Switching functionality with traditional Layer 2 networks and VPLS networks
- Inter-VXLAN routing and VXLAN-only bridging domain with IRB
- Virtual switches
- VXLAN with VRF functionality
- Configurable load balancing
- Statistics for remote VTEP

Starting in Junos OS Release 17.3, support is extended to VXLAN gateway implementation using an IPv6 underlay.

The following service types are supported with the IPv6 underlay support:

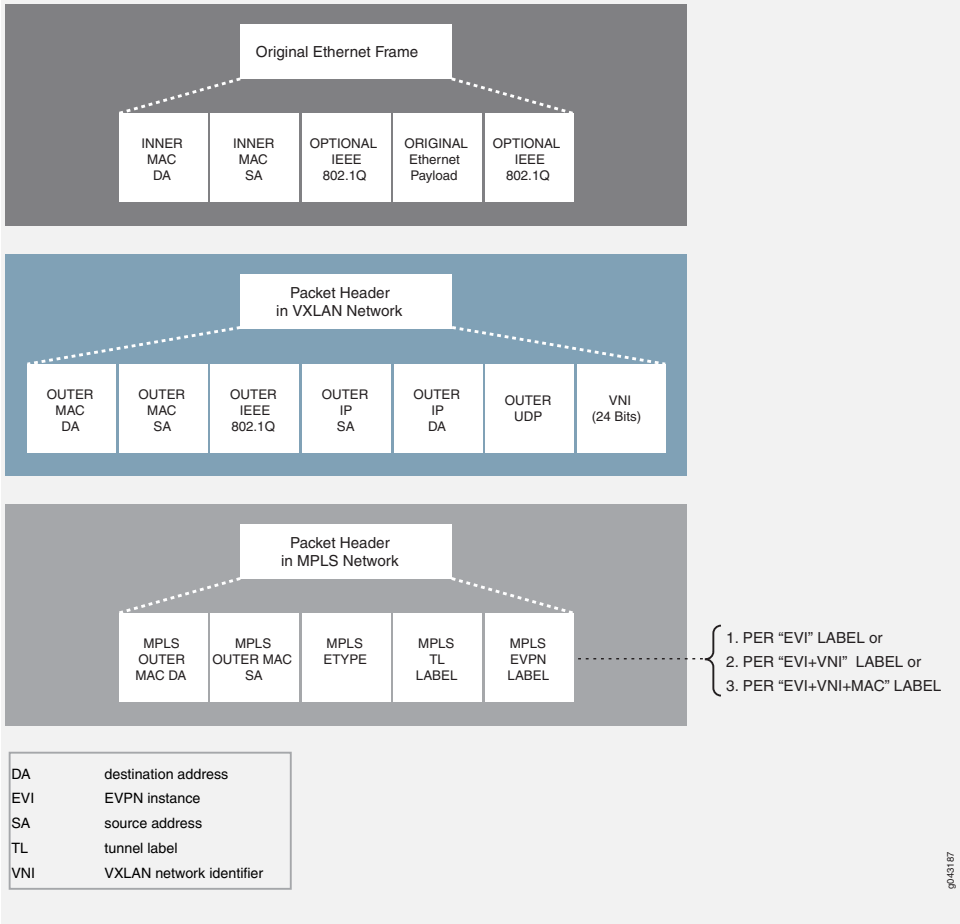
- VLAN-based service
- VLAN-bundle service
- Port-based service
- VLAN-aware service

Both IPv4 and IPv6 EVPN-VXLAN underlays support the Type 2 MAC address with IP address advertisement and the proxy MAC address with IP address advertisement.

EVPN-VXLAN Packet Format

The EVPN-VXLAN packet format is shown in Figure 17 on page 254.

Figure 17: EVPN-VXLAN Packet Format



Release History Table

Release	Description
18.3R1	(QFX10000 switches) Starting with Junos OS Release 18.3R1, you can configure a firewall filter on an IRB interface in an EVPN-VXLAN topology. Firewall filters can only be configured on the IRB interface after the VXLAN header is stripped by the VXLAN tunnel endpoint (VTEP).
18.2R1	(QFX5100, QFX5100 Virtual Chassis, and QFX5110 switches) Starting with Junos OS Release 18.2R1, you can configure firewall filters and policers on VXLAN traffic in an EVPN topology.
17.3R1	Starting with Junos OS Release 17.3R1, EX9200 switches also support EVPN-VXLAN with Contrail.

Related Documentation

- EVPN Multihoming Overview on page 29
- Understanding EVPN Pure Type-5 Routes on page 15
- Understanding VXLANs on page 260
- EVPN-over-VXLAN Supported Functionality on page 255
- EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches on page 270

EVPN-over-VXLAN Supported Functionality

The following functionality is supported for EVPN-over-VXLAN data plane encapsulation:

- VXLAN Encapsulation on page 255
- EVPN BGP Routes and Attributes on page 256
- EVPN Multihoming Procedure on page 256
- Next-Hop Forwarding on page 257
- Control Plane MAC Learning Method on page 258
- Contrail vRouters and the L3-VRF Table on page 258
- Designated Forwarder Election on page 259

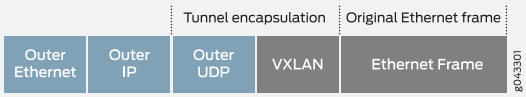
VXLAN Encapsulation

EVPN supports the VXLAN data plane encapsulation type within the same EVPN instance. To support the VXLAN encapsulation type, all EVPN PE devices specify the tunnel type as VXLAN in the BGP encapsulation extended community. The ingress PE determines which encapsulation types to use based on its own encapsulation capability and the one its EVPN remote PE device advertises.

When EVPN is used as overlay solution for the underlay IP network with VXLAN encapsulation, the data packet is encapsulated with a VXLAN header at the ingress PE device, and the data packet is de-encapsulated from the VXLAN header at the egress

PE device. Figure 18 on page 256 shows the packet format when a packet is forwarded in the core through the underlay IP network with VXLAN encapsulation:

Figure 18: Underlay IP Network with VXLAN Encapsulation



EVPN BGP Routes and Attributes

EVPN BGP routes and attributes support EVPN-over-VXLAN data plane encapsulation and are affected as follows:

- By attaching the BGP encapsulation extended community attribute with tunnel encapsulation type VXLAN in the EVPN MAC routes, the egress PE device advertises the EVPN inclusive multicast route and EVPN per EVI route autodiscovery.
- The Ethernet tag fields for all EVPN BGP routes are set to zero to support VXLAN network identifier (VNI)-based mode only.
- The VNI, also referred to as the VNI field in the EVPN overlay, is placed in the MPLS fields for the EVPN MAC route, the EVPN inclusive multicast route, and per EVPN instance autodiscovery route.
- The Ethernet segment identifier label field is set to zero for the EVPN per Ethernet segment autodiscovery route because no Ethernet segment identifier label exists for the supported VXLAN encapsulation type.
- All corresponding EVPN routes from the remote PE device are resolved over the inet.0 or :xvlan.inet.0 table instead of the inet.3 table.

EVPN Multihoming Procedure

You can multihome a bare-metal server (BMS) to a pair of Juniper Networks top-of-rack (TOR) switches, for example, QFX5100 switches, through a Layer 2 link aggregation group (LAG) interface. Because of the LAG interface, only one copy of BUM traffic is forwarded from a BMS to the core router. To prevent a Layer 2 loop and flooding of BUM traffic back to the same Ethernet segment to which the multihomed BMS is attached, the BUM traffic applies the multihomed active-active split-horizon filtering rule. To fully support the EVPN multihomed active-active mode feature, the Juniper Networks TOR switch also advertises the EVPN aliasing route to other EVPN PE devices.

The lack of MPLS label support for EVPN over IP with VXLAN data plane encapsulation causes impacts to the following EVPN multihomed active-active mode functions:

- Local Bias and Split-Horizon Filtering Rule on page 257
- Aliasing on page 257

NOTE: EVPN over VxLAN does not support active-standby mode of operation. You can only configure the active-active mode by using all-active option for the ESI interface configuration. Single-homing Ethernet segments with EVPN using VxLAN encapsulation using single-active is not supported.

Local Bias and Split-Horizon Filtering Rule

Because of the missing MPLS label, the split-horizon filtering rule for the multihomed Ethernet segment is modified and based on the IP address of the EVPN PE device instead of the MPLS Ethernet segment label. For traffic coming from the access interface, any traffic forwarding to the multihomed Ethernet segment is based on the local bias for EVPN with VXLAN data plane encapsulation. Each EVPN PE device tracks the IP address of its peer multihomed EVPN PE device for which it shares the same Ethernet segment. This tracking provides the source VXLAN virtual tunnel endpoint (VTEP) IP address [outer Session Initiation Protocol (SIP)] for each VXLAN packet received from the other EVPN PE device. The split-horizon filtering rule is enforced on both ingress and egress PE devices for multi-destination traffic:

- Ingress PE—Responsible for forwarding multi-destination packets coming from any of its directly-attached access interfaces to its remaining associated multihomed Ethernet segments, regardless of the subject ingress PE device's designated forwarder (DF) election status.
- Egress PE—Allows no forwarding of any multi-destination packets to the same multihomed Ethernet segment to which an egress PE shares with its ingress PE device, regardless of the subject egress PE device's DF election status.

Aliasing

When you connect a BMS to a pair of Juniper Networks TOR switches through a LAG interface bundle, only one of the switches can learn the local MAC address. To support the load balancing of known unicast traffic coming from the VXLAN to the BMS between the switches, the EVPN PE device on the switch must advertise EVPN per EVPN instance autodiscovery route. This advertisement signals to the remote EVPN PE devices that the MAC learned from the multihomed Ethernet segment from its peer multihomed PE device is also reachable. For VXLAN encapsulation, there is no change to the EVPN procedure when providing the EVPN aliasing function.

Next-Hop Forwarding

The Layer 2 address learning daemon (l2ald) creates the VXLAN encapsulation composite next-hop at ingress, and the VXLAN de-encapsulation composite next-hop at the egress. The VXLAN encapsulation composite next-hop forwards Layer 2 unicast traffic to the remote PE device with the VXLAN encapsulation. If multiple paths are available to reach a remote MAC (as with the multihomed EVPN active-active case), the routing protocol daemon (rpd) informs the l2ald of all the remote VTEP IP addresses associated with the remote MAC. The l2ald is responsible for building an equal-cost multipath (ECMP) next hop for the remote MAC. The VXLAN de-encapsulation composite next-hop

de-encapsulates the VXLAN tunnel header at the egress and then forwards the traffic to the BMS.

For known unicast traffic:

- At ingress, the rpd does not need to add a label route in the mpls.0 table.
- At egress, the rpd does not need to add a label route to point to the table next-hop in the mpls.0 table.

Control Plane MAC Learning Method

A unique characteristic of EVPN is that MAC address learning between PE devices occurs in the control plane. The local PE router detects a new MAC address from a CE device and then, using MP-BGP, advertises the address to all the remote PE devices. This method differs from existing Layer 2 VPN solutions such as VPLS, which learn by flooding unknown unicast in the data plane. This control plane MAC learning method is a crucial component of the features and benefits that EVPN provides. Because MAC learning is handled in the control plane, EVPN has the flexibility to support different data plane encapsulation technologies between PE devices. This flexibility is important because not every backbone network may be running MPLS, especially in enterprise networks.

For control plane remote MAC learning, because the l2ald creates the VXLAN encapsulation composite next-hop and VXLAN de-encapsulation composite next-hop, the rpd no longer creates an indirect next-hop used in the MAC forwarding table. The rpd relies on the existing mechanism to inform the l2ald of the remote MAC addresses learned from the control plane. Instead of informing the l2ald about the VLAN ID and the indirect next-hop index used by the remote MAC in the MAC FIB table, the rpd informs the l2ald about the remote VTEP IP address and VXLAN network identifier. The control plane remote MAC learned route points to VXLAN encapsulation composite next-hop, or an ECMP next-hop created by the l2ald in the MAC forwarding table.

For multihomed EVPN active-active, a pair of remote VTEP IP addresses are associated with the remote MAC address. The remote VTEP IP addresses are obtained from the MAC route received either from the remote PE device, or from the aliasing route from the remote PE device. When a remote PE device withdraws a MAC route or the aliasing route for the Ethernet segment from where the MAC learned, the rpd alerts the l2ald about the change to the pair of remote VTEP IP addresses accordingly. As a result, the l2ald updates the uni-list next-hop built for this MAC. When both remote MAC routes and its associated aliasing route are withdrawn or become unresolved, the rpd informs the l2ald about the deletion of this remote MAC and the l2ald withdraws this MAC from the MAC forwarding table.

Contrail vRouters and the L3-VRF Table

The Contrail virtualization software creates virtual networks (VNs) associated with routes in a Layer 3 virtual routing and forwarding (L3-VRF) table.

The following are the associated routes:

Subnet Routes for an IRB Interface

For each pair of MAC-(virtual routing and forwarding) VRF and L3-VRF tables created for a virtual network on an MX Series router, a corresponding IRB interface is associated with the MAC-VRF and L3-VRF table pair. The L3-VRF table on the MX Series router has the subnet route from the IRB interface associated with its local virtual networks, as well as, all subnet routes of the IRB interfaces associated with other virtual networks within a data center provided by the Junos OS routing leaking feature. The MX Series routers advertise these subnet routes through MP-BGP to the Contrail control nodes. As a result, the L3-VRF table n the Contrail vRouter contains the same set of subnet routes for the IRB interfaces in its IP FIB, and the subnet routes point their next hops to the MX Series routers.

Virtual Machine Host Routes

The Contrail vRouters support proxy ARP and advertise the IP address with the EVPN MAC route for its virtual machine (VM). For both Contrail vRouters and MX Series routers, the L3-VRF table for a virtual network contains all of the VM host routes for those VMs that reside in the same virtual network, as well as, routes in all other virtual networks. intra- virtual network and inter- virtual network traffic between the VMs is forwarded directly at Layer 3 between the Contrail vRouters.

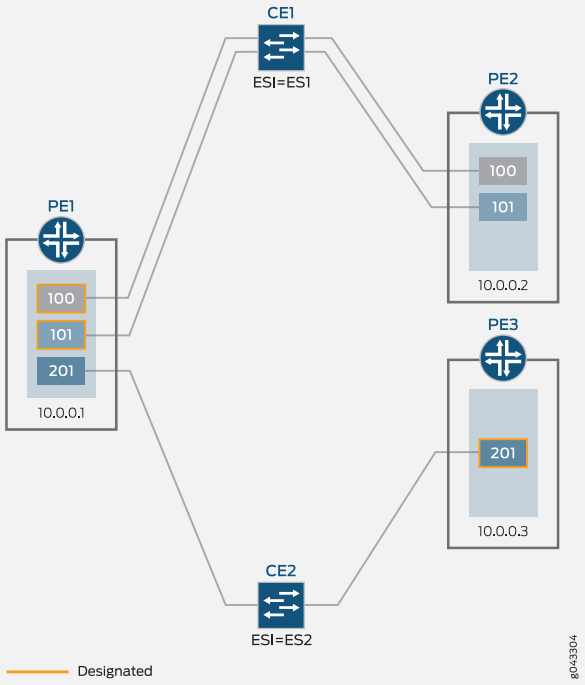
Bare-Metal Server Host Routes

A Juniper Networks TOR switch, for example, a QFX5100 switch, does not advertise the IP address with the EVPN MAC route for the bare-metal server (BMS) to which it is attached. As a result of the EVPN MAC route advertisement from the switch, no BMS host route is installed in the L3-VRF table on Contrail vRouters and MX Series routers. However, ARP routes for the BMSs are installed in the kernel on the MX Series router if the MX Series router receives ARP responses from the BMSs.

Designated Forwarder Election

To provide better load balance and more flexible topology, the election of the designated forwarder is determined by selecting the minimum VLAN ID or VXLAN network ID for each Ethernet segment instead of selecting based on the EVPN instance.
Figure 19 on page 260 shows a sample designated forwarder election topology.

Figure 19: Designated Forwarder Election Topology



CE device (CE1) has an Ethernet segment identifier value configured equal to ES1 and is connected to both PE1 and PE2 devices, with configured VLANs 100 and 101. Router PE1 is elected as the designated forwarder for the two VLANs.

CE device (CE2) has an Ethernet segment identifier value configured equal to ES2 and is connected to both PE1 and PE3 devices, with configured VLAN 201. Router PE3 is elected as the designated forwarder for this VLAN.

The Ethernet tag ID can be either of the following:

- VLAN ID (for EVPN-MPLS)
- VXLAN network ID (for EVPN-VXLAN)

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches on page 270
- Using a Default Layer 3 Gateway to Route Traffic Between Virtual Networks in an EVPN-VXLAN Topology on page 293

Understanding VXLANs

Virtual Extensible LAN protocol (VXLAN) technology allows networks to support more VLANs. According to the IEEE 802.1Q standard, traditional VLAN identifiers are 12 bits

long—this naming limits networks to 4094 VLANs. The VXLAN protocol overcomes this limitation by using a longer logical network identifier that allows more VLANs and, therefore, more logical network isolation for large networks such as clouds that typically include many virtual machines.

VXLAN Benefits

VXLAN technology allows you to segment your networks (as VLANs do), but it provides benefits that VLANs cannot. Here are the most important benefits of using VXLANs:

- You can theoretically create as many as 16 million VXLANs in an administrative domain (as opposed to 4094 VLANs on a Juniper Networks device).
  - MX Series routers and EX9200 switches support as many as 32,000 VXLANs, 32,000 multicast groups, and 8000 virtual tunnel endpoints (VTEPs). This means that VXLANs based on MX Series routers provide network segmentation at the scale required by cloud builders to support very large numbers of tenants.
  - QFX10000 Series switches support 4000 VXLANs and 2000 remote VTEPs.
  - QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches support 4000 VXLANs, 4000 multicast groups, and 2000 remote VTEPs.
- You can enable migration of virtual machines between servers that exist in separate Layer 2 domains by tunneling the traffic over Layer 3 networks. This functionality allows you to dynamically allocate resources within or between data centers without being constrained by Layer 2 boundaries or being forced to create large or geographically stretched Layer 2 domains.

Using VXLANs to create smaller Layer 2 domains that are connected over a Layer 3 network means that you do not need to use Spanning Tree Protocol (STP) to converge the topology but can use more robust routing protocols in the Layer 3 network instead. In the absence of STP, none of your links are blocked, which means you can get full value from all the ports that you purchase. Using routing protocols to connect your Layer 2 domains also allows you to load-balance the traffic to ensure that you get the best use

of your available bandwidth. Given the amount of east-west traffic that often flows within or between data centers, maximizing your network performance for that traffic is very important.

The video *Why Use an Overlay Network in a Data Center?* presents a brief overview of the advantages of using VXLANs.

Video: Why Use an Overlay Network in a Data Center?

How Does VXLAN Work?

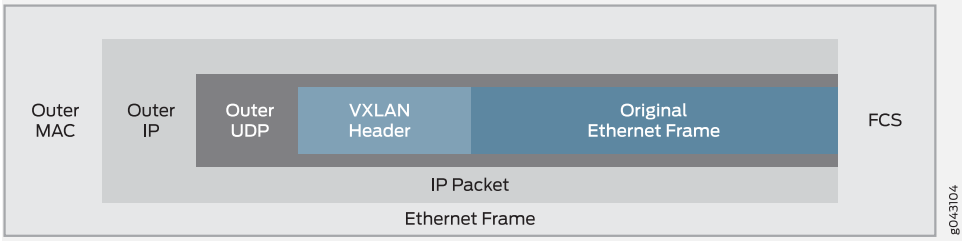
VXLAN is often described as an overlay technology because it allows you to stretch Layer 2 connections over an intervening Layer 3 network by encapsulating (tunneling) Ethernet frames in a VXLAN packet that includes IP addresses. Devices that support VXLANs are called *virtual tunnel endpoints (VTEPs)*—they can be end hosts or network switches or routers. VTEPs encapsulate VXLAN traffic and de-encapsulate that traffic when it leaves the VXLAN tunnel. To encapsulate an Ethernet frame, VTEPs add a number of fields, including the following fields:

- Outer media access control (MAC) destination address (MAC address of the tunnel endpoint VTEP)
- Outer MAC source address (MAC address of the tunnel source VTEP)
- Outer IP destination address (IP address of the tunnel endpoint VTEP)
- Outer IP source address (IP address of the tunnel source VTEP)
- Outer UDP header
- A VXLAN header that includes a 24-bit field—called the *VXLAN network identifier (VNI)*—that is used to uniquely identify the VXLAN. The VNI is similar to a VLAN ID, but having 24 bits allows you to create many more VXLANs than VLANs.

NOTE: **Because VXLAN adds 50 to 54 bytes of additional header information to the original Ethernet frame, you might want to increase the MTU of the underlying network. In this case, configure the MTU of the physical interfaces that participate in the VXLAN network, not the MTU of the logical VTEP source interface, which is ignored.**

Figure 20 on page 263 shows the VXLAN packet format.

Figure 20: VXLAN Packet Format



VXLAN Implementation Methods

Junos OS supports implementing VXLANs in the following environments:

- Manual VXLAN—In this environment, a Juniper Networks device acts as a transit device for downstream devices acting as VTEPs, or a gateway that provides connectivity for downstream servers that host virtual machines (VMs), which communicate over a Layer 3 network. In this environment, software-defined networking (SDN) controllers are not deployed.

  NOTE: QFX10000 switches do not support manual VXLANs.

- OVSDB-VXLAN—In this environment, SDN controllers use the Open vSwitch Database (OVSDB) management protocol to provide a means through which controllers (such as a VMware NSX or Juniper Networks Contrail controller) and Juniper Networks devices that support OVSDB can communicate.
- EVPN-VXLAN—In this environment, Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical servers and VMs) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network, and VXLAN creates the data plane for the Layer 2 overlay network.

Using QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches with VXLANs

You can configure the switches to perform all of the following roles:

- In an environment without an SDN controller, act as a transit Layer 3 switch for downstream hosts acting as VTEPs. In this configuration, you do not need to configure any VXLAN functionality on the switch. You do need to configure IGMP and PIM so that the switch can form the multicast trees for the VXLAN multicast groups. (See Manual VXLANs Require PIM on page 266 for more information.)
- In an environment with or without an SDN controller, act as a Layer 2 gateway between virtualized and nonvirtualized networks in the same data center or between data centers. For example, you can use the switch to connect a network that uses VXLANs to one that uses VLANs.

- Act as a Layer 2 gateway between virtualized networks in the same or different data centers and allow virtual machines to move (VMotion) between those networks and data centers. For example, if you want to allow VMotion between devices in two different networks, you can create the same VLAN in both networks and put both devices on that VLAN. The switches connected to these devices, acting as VTEPs, can map that VLAN to the same VXLAN, and the VXLAN traffic can then be routed between the two networks.
- (QFX5110 switches with EVPN-VXLAN) Act as a Layer 3 gateway to route traffic between different VXLANs in the same data center.
- (QFX5110 switches with EVPN-VXLAN) Act as a Layer 3 gateway to route traffic between different VXLANs in different data centers over a WAN or the Internet using standard routing protocols or virtual private LAN service (VPLS) tunnels.

NOTE: If you want a QFX5110 switch to be a Layer 3 VXLAN gateway in an EVPN-VXLAN environment, you must configure integrated routing and bridging (IRB) interfaces to connect the VXLANs, just as you do if you want to route traffic between VLANs.

Because the additional headers add 50 to 54 bytes, you might need to increase the MTU on a VTEP to accommodate larger packets. For example, if the switch is using the default MTU value of 1514 bytes and you want to forward 1500-byte packets over the VXLAN, you need to increase the MTU to allow for the increased packet size caused by the additional headers.

Changing the UDP Port on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches

Starting with Junos OS Release 14.1X53-D25 on QFX5100 switches, Junos OS Release 15.1X53-D210 on QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 on QFX5210 switches, and Junos OS Release 18.2R1 on EX4600 switches, you can configure the UDP port used as the destination port for VXLAN traffic. To configure the VXLAN destination port to be something other than the default UDP port of 4789, enter the following statement:

set protocols l2-learning destination-udp-port ***port-number***

The port you configure will be used for all VXLANs configured on the switch.

NOTE: If you make this change on one switch in a VXLAN, you must make the same change on all the devices that terminate the VXLANs configured on your switch. If you do not do so, traffic will be disrupted for all the VXLANs configured on your switch. When you change the UDP port, the previously learned remote VTEPs and remote MACs are lost and VXLAN traffic is disrupted until the switch relearns the remote VTEPs and remote MACs.

Controlling Transit Multicast Traffic on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches

When the switch acting as a VTEP receives a broadcast, unknown unicast, or multicast packet, it performs the following actions on the packet:

1. It de-encapsulates the packet and delivers it to the locally attached hosts.
2. It then adds the VXLAN encapsulation again and sends the packet to the other VTEPs in the VXLAN.

These actions are performed by the loopback interface used as the VXLAN tunnel address and can, therefore, negatively impact the bandwidth available to the VTEP. Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 for QFX5210 switches, and Junos OS Release 18.2R1 for EX4600 switches, if you know that there are no multicast receivers attached to other VTEPs in the VXLAN that want traffic for a specific multicast group, you can reduce the processing load on the loopback interface by entering the following statement:

set protocols l2-learning disable-vxlan-multicast-transit vxlan-multicast-group *multicast-group*

In this case, no traffic will be forwarded for the specified group but all other multicast traffic will be forwarded. If you do not want to forward any multicast traffic to other VTEPs in the VXLAN, enter the following statement:

set protocols l2-learning disable-vxlan-multicast-transit vxlan-multicast-group all

Using an MX Series Router, EX9200 Switch, or QFX10000 Switch as a VTEP

You can configure an MX Series router, EX9200 switch, or QFX10000 switch to act as a VTEP and perform all of the following roles:

- Act as a Layer 2 gateway between virtualized and nonvirtualized networks in the same data center or between data centers. For example, you can use an MX Series router to connect a network that uses VXLANs to one that uses VLANs.
- Act as a Layer 2 gateway between virtualized networks in the same or different data centers and allow virtual machines to move (VMotion) between those networks and data centers.
- Act as a Layer 3 gateway to route traffic between different VXLANs in the same data center.
- Act as a Layer 3 gateway to route traffic between different VXLANs in different data centers over a WAN or the Internet using standard routing protocols or virtual private LAN service (VPLS) tunnels.

NOTE: If you want one of the devices described in this section to be a VXLAN Layer 3 gateway, you must configure integrated routing and bridging (IRB) interfaces to connect the VXLANs, just as you do if you want to route traffic between VLANs.

Manual VXLANs Require PIM

In an environment with a controller (such as a VMware NSX or Juniper Networks Contrail controller), you can provision VXLANs on a Juniper Networks device. A controller also provides a control plane that VTEPs use to advertise their reachability and learn about the reachability of other VTEPs. You can also manually create VXLANs on Juniper Networks devices instead of using a controller. If you use this approach, you must also configure Protocol Independent Multicast (PIM) on the VTEPs so that they can create VXLAN tunnels between themselves.

You must also configure each VTEP in a given VXLAN to be a member of the same multicast group. (If possible, you should assign a different multicast group address to each VXLAN, although this is not required. Multiple VXLANs can share the same multicast group.) The VTEPs can then forward ARP requests they receive from their connected hosts to the multicast group. The other VTEPs in the group de-encapsulate the VXLAN information, and (assuming they are members of the same VXLAN) they forward the ARP request to their connected hosts. When the target host receives the ARP request, it responds with its MAC address, and its VTEP forwards this ARP reply back to the source VTEP. Through this process, the VTEPs learn the IP addresses of the other VTEPs in the VXLAN and the MAC addresses of the hosts connected to the other VTEPs.

The multicast groups and trees are also used to forward broadcast, unknown unicast, and multicast (BUM) traffic between VTEPs. This prevents BUM traffic from being unnecessarily flooded outside the VXLAN.

NOTE: Multicast traffic that is forwarded through a VXLAN tunnel is sent only to the remote VTEPs in the VXLAN. That is, the encapsulating VTEP does not copy and send copies of the packets according to the multicast tree—it only forwards the received multicast packets to the remote VTEPs. The remote VTEPs de-encapsulate the encapsulated multicast packets and forward them to the appropriate Layer 2 interfaces.Junos OS Release 18.1R1 for QFX5210 switches

Load Balancing VXLAN Traffic

On QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches, the Layer 3 routes that form VXLAN tunnels use per-packet load balancing by default, which means that load balancing is implemented if there are ECMP paths to the remote VTEP. This is different from normal routing behavior in which per-packet load balancing is not used by default. (Normal routing uses per-prefix load balancing by default.)

The source port field in the UDP header is used to enable ECMP load balancing of the VXLAN traffic in the Layer 3 network. This field is set to a hash of the inner packet fields, which results in a variable that ECMP can use to distinguish between tunnels (flows). (None of the other fields that flow-based ECMP normally uses are suitable for use with VXLANs. All tunnels between the same two VTEPs have the same outer source and destination IP addresses, and the UDP destination port is set to port 4789 by definition. Therefore, none of these fields provide a sufficient way for ECMP to differentiate flows.)

VLAN IDs for VXLANs

When configuring a VLAN ID for a VXLAN on any Juniper Networks device that supports VXLANs except QFX10000 switches, we strongly recommend using a VLAN ID of 3 or higher. If you use a VLAN ID of 1 or 2, replicated broadcast, multicast, and unknown unicast (BUM) packets for these VXLANs might be untagged, which in turn might result in the packets being dropped by a device that receives the packets.

Release History Table

Release	Description
14.1X53-D30	Starting with Junos OS Release 14.1X53-D30 for QFX5100 switches, Junos OS Release 15.1X53-D210 for QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 for QFX5210 switches, and Junos OS Release 18.2R1 for EX4600 switches, if you know that there are no multicast receivers attached to other VTEPs in the VXLAN that want traffic for a specific multicast group, you can reduce the processing load on the loopback interface
14.1X53-D25	Starting with Junos OS Release 14.1X53-D25 on QFX5100 switches, Junos OS Release 15.1X53-D210 on QFX5110 and QFX5200 switches, Junos OS Release 18.1R1 on QFX5210 switches, and Junos OS Release 18.2R1 on EX4600 switches, you can configure the UDP port used as the destination port for VXLAN traffic.

Related Documentation

- *Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- *OVSDB Support on Juniper Networks Devices*
- *mtu*

VXLAN Constraints on QFX Series and EX4600 Switches

When configuring Virtual Extensible LANs (VXLANs) on QFX Series and EX4600 switches, be aware of the constraints described in the following sections. In these sections, “Layer 3 side” refers to a network-facing interface that performs VXLAN encapsulation and de-encapsulation, and “Layer 2 side” refers to a server-facing interface that is a member of a VLAN that is mapped to a VXLAN.

- VXLAN Constraints on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches on page 268
- VXLAN Constraints on QFX10000 Switches on page 269

VXLAN Constraints on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 Switches

- (QFX5100 switches only) You can use VXLANs on a Virtual Chassis or Virtual Chassis Fabric (VCF) if all of the members are supported QFX5100 switches. You cannot use VXLANs if any of the members is not a supported QFX5100 switch.
- (EX4600 switches only) You can use VXLANs on a Virtual Chassis if all of the members are supported EX4600 switches. You cannot use VXLANs if any of the members is not a supported EX4600 switch.
- Routing instance constraints:
  - (QFX5100, QFX5200, QFX5210, and EX4600 switches) VXLAN configuration is supported only in the default routing instance.
  - (QFX5110 switches) VXLAN configuration is supported only in a single virtual switching instance and multiple VPN routing and forwarding (VRF) instances.
- (QFX5100, QFX5200, QFX5210, and EX4600 switches) Routing traffic between different VXLANs is not supported.
- (QFX5110 switches only) Routing traffic between a VXLAN and a VLAN is not supported.
- (QFX5110 switches only) Integrated routing and bridging (IRB) interfaces used in EVPN-VXLAN overlay networks do not support routing protocols such as OSPF, IS-IS, and BGP.
- A physical interface cannot be a member of a VLAN and a VXLAN. That is, an interface that performs VXLAN encapsulation and de-encapsulation cannot also be a member of a VLAN. For example, if a VLAN that is mapped to a VXLAN is a member of trunk port xe-0/0/0, any other VLAN that is a member of xe-0/0/0 must also be assigned to a VXLAN.
- Multichassis link aggregation groups (MC-LAGs) are not supported with VXLAN.

  NOTE: **In an EVPN-VXLAN environment, EVPN multihoming active-active mode is used instead of MC-LAG for redundant connectivity between hosts and leaf devices.**

- IP fragmentation and defragmentation are not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:
  - (QFX5100, QFX5200, QFX5210, and EX4600 switches) IGMP snooping with EVPN-VXLAN.
  - Redundant trunk groups (RTGs).
  - The ability to shut down a Layer 2 interface or temporarily disable the interface when a storm control level is exceeded is not supported.
  - STP (any variant).
- Access port security features are not supported with VXLAN. For example, the following features are not supported:

- DHCP snooping.
- Dynamic ARP inspection.
- MAC limiting and MAC move limiting.

NOTE: An exception to this constraint is that MAC limiting is supported on OVSDB-managed interfaces in an OVSDB-VXLAN environment with Contrail controllers. For more information, see *Features Supported on OVSDB-Managed Interfaces*.

- Ingress node replication is not supported in the following cases:
  - When PIM is used for the control plane (manual VXLAN).
  - When an SDN controller is used for the control plane (OVSDB-VXLAN).

  Ingress node replication is supported with EVPN-VXLAN.
- PIM-BIDIR and PIM-SSM are not supported with VXLANs.
- If you configure a port-mirroring instance to mirror traffic exiting from an interface that performs VXLAN encapsulation, the source and destination MAC addresses of the mirrored packets are invalid. The original VXLAN traffic is not affected.
- When configuring a VLAN ID for a VXLAN, we strongly recommend using a VLAN ID of 3 or higher. If you use a VLAN ID of 1 or 2, replicated broadcast, multicast, and unknown unicast (BUM) packets for these VXLANs might be untagged, which in turn might result in the packets being dropped by a device that receives the packets.
- (QFX5110 switches only) VLAN firewall filters are not supported on IRB interfaces on which EVPN-VXLAN is enabled.
- (QFX5100, QFX5100 Virtual Chassis, and QFX5110 switches) Firewall filters and policers are not supported on transit traffic on which EVPN-VXLAN is enabled. They are supported only in the ingress direction on CE-facing interfaces.
- (QFX5100, QFX5100 Virtual Chassis, and QFX5110 switches) For IRB interfaces in an EVPN-VXLAN one-layer IP fabric, firewall filtering and policing is supported only at the ingress point of non-encapsulated frames routed through the IRB interface.

VXLAN Constraints on QFX10000 Switches

- MC-LAGs are not supported with VXLAN.

NOTE: In an EVPN-VXLAN environment, EVPN multihoming active-active mode is used instead of MC-LAG for redundant connectivity between hosts and leaf devices.

- IP fragmentation is not supported on the Layer 3 side.
- The following features are not supported on the Layer 2 side:

  - IGMP snooping with EVPN-VXLAN in Junos OS Releases before Junos OS Release 17.2R1.
  - STP (any variant).
- Access port security features are not supported with VXLAN. For example, the following features are not supported:
  - DHCP snooping.
  - Dynamic ARP inspection.
  - MAC limiting and MAC move limiting.
- Ingress node replication is not supported when an SDN controller is used for the control plane (OVSDB-VXLAN). Ingress node replication is supported for EVPN-VXLAN.
- QFX10000 switches that are deployed in an EVPN-VXLAN environment do not support an IPv6 physical underlay network.
- When the next-hop database on a QFX10000 switch includes next hops for both the underlay network and the EVPN-VXLAN overlay network, the next hop to a VXLAN peer cannot be an Ethernet segment identifier (ESI) or a virtual tunnel endpoint (VTEP) interface.
- IRB interfaces used in EVPN-VXLAN overlay networks do not support routing protocols such as OSPF, IS-IS, and BGP.
- VLAN firewall filters applied to IRB interfaces on which EVPN-VXLAN is enabled.
- Filter-based forwarding (FBF) is not supported on IRB interfaces used in an EVPN-VXLAN environment.

Related Documentation

- Understanding VXLANs on page 260
- *Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- *Manually Configuring VXLANs on QFX Series and EX4600 Switches*

EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches

This topic provides information about configuring Ethernet VPN (EVPN) with Virtual Extensible Local Area Networks (VXLAN) data plane encapsulation on QFX5100, QFX5110, QFX5200, QFX5210, and EX4600 switches.

The configuration statements described in this topic are in the **switch-options** and **protocols evpn** hierarchy levels.

CLI Statement	Description
route-distinguisher	Specifies an identifier attached to a route—this enables you to distinguish to which VPN or VPLS the route belongs. Each routing instance must have a unique route distinguisher associated with it. The route distinguisher is used to place bounds around a VPN so that the same IP address prefixes can be used in different VPNs without having them overlap.
vrf-target	Specifies a VRF target community. In effect, this statement configures a single policy for import and a single policy for export to replace the per-VRF policies for every community. **import**, **export**, and **auto** options are available.
vrf-import	Specifies how routes are imported into the VRF table of the local provider edge (PE) router or switch from the remote PE.
vrf-export	Specifies how routes are exported from the local PE router's VRF table to the remote PE router.
designated-forwarder-election-hold-time	Establishes when a designated forwarder is required for customer edge (CE) devices that are multihomed to more than one provider edge (PE) device. Without a designated forwarder, multihomed hosts receive duplicate packets. Designated forwarders are chosen for an Ethernet segment identifier (ESI) based on type-4 route advertisements.
encapsulation vxlan	Configures a VXLAN encapsulation type.
extended-vni-list and extended-vni-all	Establishes which VXLAN virtual network identifiers are designated as part of the virtual switch instance.
multicast-mode	Configures the multicast server mode for delivering traffic and packets for EVPN.

CLI Statement	Description
vni-options	Configures different route targets for each VXLAN virtual network identifier instance under vni-options.
show route table	Displays both imported EVPN routes, and export/import EVPN routes for the default-switch routing instance.
show configuration protocols evpn	Displays results of the extended-vni-list and vni-options statements.

Related Documentation

- EVPN Multihoming Overview on page 29
- Understanding VXLANs on page 260
- EVPN-over-VXLAN Supported Functionality on page 255
- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373

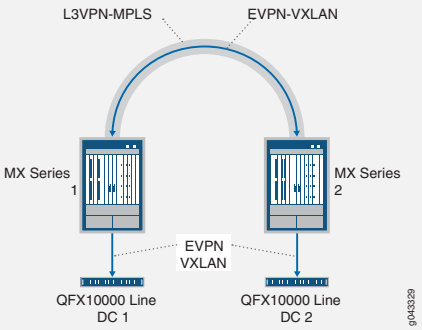
Implementing EVPN-VXLAN for Data Centers

Although there are various Data center interconnect (DCI) technologies available, EVPN has an added advantage over other MPLS technologies because of its unique features, such as active/active redundancy, aliasing, and mass MAC withdrawal. To provide a DCI solution, VXLAN is integrated with EVPN.

There are different options for using EVPN-VXLAN with DCI:

- DCI can connect multiple data centers in your WAN using MX Series edge routers with a Layer 3 VPN MPLS network between them. QFX10000 switches start and stop the VXLAN tunnel. This option requires no changes to your WAN.

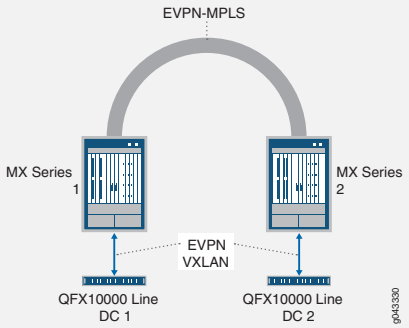
Figure 21: DCI Option: Layer 3 VPN-MPLS



- A second option connects multiple data centers in your WAN using either MX Series edge routers or supported QFX Series switches with an EVPN-MPLS network between them. This option uses an EVPN data plane and an MPLS control plane and requires

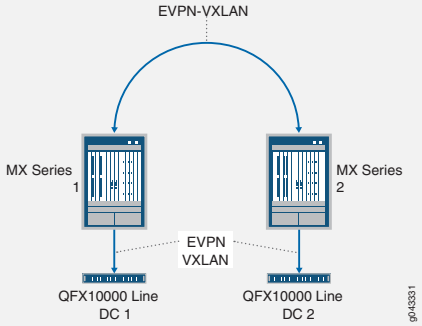
changes to your WAN. You must change your LAN architecture to natively support EVPN, and you must implement EVPN stitching between each MX router/QFX Series switch and the corresponding QFX10000 switch. For details about releases where QFX Series switches are supported, see https://pathfinder.juniper.net/feature-explorer and then search on EVPN.

Figure 22: DCI Option: EVPN-MPLS



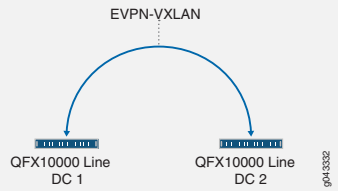
- You can also tunnel two branch locations across the Internet. In this case, implementation requires neither a traditional WAN nor MPLS. This method can use the Internet or an IP tunnel, where VXLAN rides on top of IP and EVPN is used throughout.

Figure 23: DCI Option: EVPN-VXLAN over the Internet



- If you do not have a branch router or a peering router, you can simply connect the data centers directly and EVPN is again used natively throughout. This implementation requires neither a traditional WAN nor MPLS, but you typically need a dark fiber connection.

Figure 24: DCI Option: Layer 3 VPN-MPLS Direct Connection



You can alternately create an EVPN-VXLAN fabric internally in the data center using bare-metal servers and/or virtual servers and using OpenClos for management. Here you also use VXLAN L2 gateways and L3 gateways on switches such as a QFX10000 switch. The underlying fabric is built on BGP.

EVPN-VXLAN uses both routers and switches—the configurations are the same for both devices but they are located in different areas of the Junos OS CLI. MX Series routers are configured under a routing instance with the instance type **virtual switch**. QFX Series switches are configured under global **switching-options** and global **protocol evpn**. See Table 10 on page 274 for a list of CLI commands used by EVPN-VXLAN.

Table 10: CLI Commands for EVPN-VXLAN

Function	CLI Command
Specifies an identifier attached to a route. This enables you to distinguish to which VPN or VPLS the route belongs. Each routing instance must have a unique route distinguisher (RD) associated with it. The RD is used to place bounds around a VPN so that the same IP address prefixes can be used in different VPNs without having them overlap.	route-distinguisher
Specifies a VRF target community. In effect, this statement configures a single policy for import and a single policy for export to replace the per-VRF policies for every community. The options **import** and **export** apply to both routers and QFX Series switches. The option **auto** applies to QFX Series switches only.	vrf-target
Specifies how routes are imported into the VRF table of the local PE router or switch from the remote PE router.	vrf-import
Specifies how routes are exported from the local PE router's VRF table to the remote PE router.	vrf-export
A designated forwarder (DF) is required when CEs are multihomed to more than one PE. Without a designated forwarder, multihomed hosts would receive duplicate packets. Designated forwarders are chosen for an Ethernet segment identifier (ESI) based on type-4 route advertisements.	designated-forwarder-election-hold-time
Configures a logical link-layer encapsulation type.	*encapsulation*
Establishes which VXLAN virtual network identifiers (VNIs) will be part of the EVPN-VXLAN MP-BGP domain. There are different BUM replication options available in EVPN—using **extended-vni-list** forgoes a multicast underlay in favor of EVPN-VXLAN ingress replication.	extended-vni-list

Table 10: CLI Commands for EVPN-VXLAN (continued)

Function	CLI Command
You configure different route targets (RTs) for each VNI instance under vni-options.	vni-options (QFX Series switches only)
Displays both imported EVPN routes and export/import EVPN routes for the default switch routing instances.	show route table
Displays results of the configuration commands extended-vni-list and vni-options.	*show configuration protocols evpn*

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247

PIM NSR and Unified ISSU Support for VXLAN Overview

Starting in Junos OS Release 16.2R1, Protocol Independent Multicast (PIM) nonstop active routing (NSR) support for Virtual Extensible LAN (VXLAN) is supported on MX Series routers.

The Layer 2 address learning daemon (l2ald) passes VXLAN parameters (VXLAN multicast group addresses and the source interface for a VXLAN tunnel [**vtep-source-interface**]) to the routing protocol process on the master Routing Engine. The routing protocol process forms PIM joins with the multicast routes through the pseudo-VXLAN interface based on these configuration details.

Because the l2ald daemon does not run on the backup Routing Engine, the configured parameters are not available to the routing protocol process in the backup Routing Engine when NSR is enabled. The PIM NSR mirroring mechanism provides the VXLAN configuration details to the backup Routing Engine, which enables creation of the required states. The routing protocol process matches the multicast routes on the backup Routing Engine with PIM states, which maintains the multicast routes in the Forwarding state.

In response to Routing Engine switchover, the multicast routes remain in the Forwarding state on the new master Routing Engine. This prevents traffic loss during Routing Engine switchover. When the l2ald process becomes active, it refreshes VXLAN configuration parameters to PIM.

NOTE: **For this feature, NSR support is available for VXLAN in PIM sparse mode.**

This feature does not introduce any new CLI commands. You can issue the following **show** commands on the backup Routing Engine to monitor the PIM joins and multicast routes on the backup Routing Engine:

- **show pim join extensive**
- **show multicast route extensive**

Unified ISSU Support

Starting in Junos OS Release 17.2 R1, unified in-service software upgrade is supported for VXLAN using PIM on MX Series routers. ISSU enables you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Unified ISSU is supported only on dual Routing Engine platforms. The graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) features must both be enabled. Unified ISSU allows you to eliminate network downtime, reduce operating costs, and deliver higher levels of services. See *Getting Started with Unified In-Service Software Upgrade*.

NOTE: Unified ISSU is not supported on the QFX series switches.

To enable GRES, include the **graceful-switchover** statement at the **[edit chassis redundancy]** hierarchy level.

To enable NSR, include the **nonstop-routing** statement at the **[edit routing-options]** hierarchy level and the **commit synchronize** statement at the **[edit system]** hierarchy level.

Release History Table

Release	Description
17.2R1	Starting in Junos OS Release 17.2 R1, unified in-service software upgrade is supported for VXLAN using PIM on MX Series routers.
16.2R1	Starting in Junos OS Release 16.2R1, Protocol Independent Multicast (PIM) nonstop active routing (NSR) support for Virtual Extensible LAN (VXLAN) is supported on MX Series routers.

Related Documentation

- *show pim join*
- *show multicast route*
- *Understanding Graceful Routing Engine Switchover*

Routing IPv6 Data Traffic through an EVPN-VXLAN Network with an IPv4 Underlay

QFX10000 and QFX5110 switches can function as Layer 3 VXLAN gateways in an EVPN-VXLAN overlay network. Starting with Junos OS Release 15.1X53-D30 for QFX10000 switches and Junos OS Release 18.3R1 for QFX5110 switches, the IRB interfaces on these Layer 3 VXLAN gateways can route Layer 2 or 3 data packets from one IPv6 host to another IPv6 host through an EVPN-VXLAN network with an IPv4 underlay.

Upon receipt of a Layer 2 or 3 data packet from an IPv6 host, a Layer 3 VXLAN gateway encapsulates the packet with an IPv4 outer header, thereby tunneling the packet through the IPv4 underlay network. The Layer 2 or 3 VXLAN gateway at the other end of the tunnel de-encapsulates the packet and forwards the packet towards the other IPv6 host.

The Layer 3 VXLAN gateways in the EVPN-VXLAN overlay network learn the IPv6 routes through the exchange of EVPN Type-2 and Type-5 routes.

This feature is supported in EVPN-VXLAN networks with the following IPv4 fabric architectures, which are commonly deployed within a data center:

- **IPv4 fabric scenario 1**—A two-layer IPv4 fabric in which QFX10000 or QFX5110 switches function as Layer 3 VXLAN gateways, and QFX5100 or QFX5200 switches function as Layer 2 VXLAN gateways. In this architecture, the IRB interfaces configured on the Layer 3 VXLAN gateways function in a central location in the fabric. The IRB interfaces in this IPv4 fabric comprise an overlay known as a *centrally-routed bridging overlay*.
- **IPv4 fabric scenario 2**—A one-layer IPv4 fabric in which QFX10000 or QFX5110 switches function as both Layer 2 and Layer 3 VXLAN gateways. In this architecture, the IRB interfaces configured on the Layer 3 VXLAN gateways function at the edge of the fabric. The IRB interfaces in this IPv4 fabric comprise an overlay known as an *edge-routed bridging overlay*.

This topic includes the following sections:

Benefits of Routing IPv6 Data Traffic through an EVPN-VXLAN Network With an IPv4 Underlay

Routing IPv6 data traffic through an EVPN-VXLAN network with an IPv4 underlay provides the following benefits:

- Eliminates the need to deploy an IPv6 underlay network. The supported IPv4 fabric architectures are agile enough to support both IPv4 and IPv6 data traffic needs.
- Leverages the EVPN control plane's inherent support for exchanging IPv4 and IPv6 routes.
- Does not introduce any new configuration. To set up this feature, you configure IRB interfaces with IPv6 addresses.

IPv4 Fabric Scenario 1—How to Set Up the Routing of IPv6 Data Traffic

NOTE: The focus of this section is the configuration of IRB interfaces on Layer 3 VXLAN gateways. For a more comprehensive example of configuring EVPN-VXLAN, see "Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center" on page 302.

The key to setting up this feature is the configuration of IRB interfaces on Layer 3 VXLAN gateways. In general, you configure the IRB interfaces as you would with only IPv4 hosts in the topology. However, in addition to specifying IPv4 addresses for the IRB interface and default Layer 3 gateway (virtual gateway), you also specify IPv6 addresses.

Table 11 on page 278 shows how to configure the addresses for IRB interfaces irb.100 and irb.200 on the three Layer 3 VXLAN gateways in a two-layer IPv4 fabric. Table 12 on page 280 outlines some additional required global configuration and configuration for irb.100 and irb.200.

Table 11: IPv4 Fabric Scenario 1—Sample IRB Interface Addresses

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
IRB interface irb.100				
IRB IPv4 address	192.168.100.1/24	192.168.100.2/24	192.168.100.3/24	Specify a different IPv4 address for irb.100 on each device.
IRB global IPv6 address	2001:db8::192.168.100.1/96	2001:db8::192.168.100.2/96	2001:db8::192.168.100.3/96	Specify a different IPv6 address for irb.100 on each device.
IRB link-local IPv6 address	fe80::100:00:01/96	fe80::100:00:01/96	fe80::100:00:01/96	Specify the same link-local IPv6 address for irb.100 on each device. Any packet destined to this IPv6 address is intercepted for Network Discovery Protocol (NDP) processing.
Virtual gateway IPv4 address	192.168.100.254/24	192.168.100.254/24	192.168.100.254/24	Specify the same IPv4 anycast address for the virtual gateway on each device.
Virtual gateway IPv6 address	2001:db8::192.168.100.254/96	2001:db8::192.168.100.254/96	2001:db8::192.168.100.254/96	Specify the same IPv6 anycast address for the virtual gateway on each device.

Table 11: IPv4 Fabric Scenario 1—Sample IRB Interface Addresses (continued)

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
Virtual gateway IPv4 and IPv6 MAC addresses	Method 1 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:fe Method 2 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:ff Method 3 • IPv4 MAC address: 00:00:5e:00:01:01 • IPv6 MAC address: 00:00:5e:00:02:01	Method 1 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:fe Method 2 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:ff Method 3 • IPv4 MAC address: 00:00:5e:00:01:01 • IPv6 MAC address: 00:00:5e:00:02:01	Method 1 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:fe Method 2 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:ff Method 3 • IPv4 MAC address: 00:00:5e:00:01:01 • IPv6 MAC address: 00:00:5e:00:02:01	For the default Layer 3 gateway, the IPv4 and IPv6 MAC addresses can be the same or different, as long as they are consistent across the three devices. Here are the supported options: • Method 1—you explicitly configure the same IPv4 and IPv6 MAC address on each device. • Method 2—you explicitly configure different IPv4 and IPv6 MAC addresses on each device. • Method 3—you do not explicitly configure IPv4 and IPv6 MAC addresses, and the system automatically generates 00:00:5e:00:01:01 as the IPv4 MAC address and 00:00:5e:00:02:01 as the IPv6 MAC address.
IRB interface irb.200				
IRB IPv4 address	192.168.200.1/24	192.168.200.2/24	192.168.200.3/24	Specify a different IPv4 address for irb.200 on each device.
IRB global IPv6 address	2001:db8::192.168.200.1/96	2001:db8::192.168.200.2/96	2001:db8::192.168.200.3/96	Specify a different IPv6 address for irb.200 on each device.
IRB link-local IPv6 address	fe80::200:00:01/96	fe80::200:00:01/96	fe80::200:00:01/96	Specify the same link-local IPv6 address for irb.200 on each device. Any packet destined to this IPv6 address is intercepted for NDP processing.

Table 11: IPv4 Fabric Scenario 1—Sample IRB Interface Addresses (continued)

Addresses	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3	Description
Virtual gateway IPv4 address	192.168.200.254/24	192.168.200.254/24	192.168.200.254/24	Specify the same IPv4 anycast address for the virtual gateway on each device.
Virtual gateway IPv6 address	2001:db8::192.168.200.254/96	2001:db8::192.168.200.254/96	2001:db8::192.168.200.254/96	Specify the same IPv6 anycast address for the virtual gateway on each device.
Virtual gateway IPv4 and IPv6 MAC addresses	Method 1 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:fe Method 2 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:ff Method 3 • IPv4 MAC address: 00:00:5e:00:01:01 • IPv6 MAC address: 00:00:5e:00:02:01	Method 1 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:fe Method 2 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:ff Method 3 • IPv4 MAC address: 00:00:5e:00:01:01 • IPv6 MAC address: 00:00:5e:00:02:01	Method 1 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:fe Method 2 • IPv4 MAC address: 10:00:00:00:00:fe • IPv6 MAC address: 10:00:00:00:00:ff Method 3 • IPv4 MAC address: 00:00:5e:00:01:01 • IPv6 MAC address: 00:00:5e:00:02:01	For the default Layer 3 gateway, the IPv4 and IPv6 MAC addresses can be the same or different, as long as they are consistent across the three devices. Here are the supported options: • Method 1—you explicitly configure the same IPv4 and IPv6 MAC address on each device. • Method 2—you explicitly configure different IPv4 and IPv6 MAC addresses on each device. • Method 3—you do not explicitly configure IPv4 and IPv6 MAC addresses, and the system automatically generates 00:00:5e:00:01:01 as the IPv4 MAC address and 00:00:5e:00:02:01 as the IPv6 MAC address.

Table 12: IPv4 Fabric Scenario 1—Required IRB Interface Configuration

Description	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3
Global IRB interface configuration			

Table 12: IPv4 Fabric Scenario 1—Required IRB Interface Configuration (continued)

Description	Layer 3 VXLAN Gateway 1	Layer 3 VXLAN Gateway 2	Layer 3 VXLAN Gateway 3
Specify that the IPv4 and IPv6 MAC addresses of the default Layer 3 gateway are advertised to the Layer 2 VXLAN gateways without the extended community option.	set protocols evpn default-gateway no-gateway-community	set protocols evpn default-gateway no-gateway-community	set protocols evpn default-gateway no-gateway-community
IRB interface irb.100 configuration			
Configure the Layer 3 VXLAN gateway to advertise the MAC and IP routes (MAC+IP type 2 routes) on behalf of the Layer 2 VXLAN gateways.	set interfaces irb unit 100 proxy-macip-advertisement	set interfaces irb unit 100 proxy-macip-advertisement	set interfaces irb unit 100 proxy-macip-advertisement
Enable the default Layer 3 gateway to be pinged by either IPv4 or IPv6 addresses.	set interfaces irb unit 100 virtual-gateway-accept-data set interfaces irb unit 100 family inet address 192.168.100.1/24 preferred set interfaces irb unit 100 family inet6 address 2001:db8::192.168.100.1/96 preferred	set interfaces irb unit 100 virtual-gateway-accept-data set interfaces irb unit 100 family inet address 192.168.100.2/24 preferred set interfaces irb unit 100 family inet6 address 2001:db8::192.168.100.2/96 preferred	set interfaces irb unit 100 virtual-gateway-accept-data set interfaces irb unit 100 family inet address 192.168.100.3/24 preferred set interfaces irb unit 100 family inet6 address 2001:db8::192.168.100.3/96 preferred
IRB interface irb.200 configuration			
Configure the Layer 3 VXLAN gateway to advertise the MAC and IP routes (MAC+IP type 2 routes) on behalf of the Layer 2 VXLAN gateways.	set interfaces irb unit 200 proxy-macip-advertisement	set interfaces irb unit 200 proxy-macip-advertisement	set interfaces irb unit 200 proxy-macip-advertisement
Enable the default Layer 3 gateway to be pinged by either IPv4 or IPv6 addresses.	set interfaces irb unit 200 virtual-gateway-accept-data set interfaces irb unit 200 family inet address 192.168.200.1/24 preferred set interfaces irb unit 200 family inet6 address 2001:db8::192.168.200.1/96 preferred	set interfaces irb unit 200 virtual-gateway-accept-data set interfaces irb unit 200 family inet address 192.168.200.2/24 preferred set interfaces irb unit 200 family inet6 address 2001:db8::192.168.200.2/96 preferred	set interfaces irb unit 200 virtual-gateway-accept-data set interfaces irb unit 200 family inet address 192.168.200.3/24 preferred set interfaces irb unit 200 family inet6 address 2001:db8::192.168.200.3/96 preferred

IPv4 Fabric Scenario 2—How to Set Up the Routing of IPv6 Data Traffic

NOTE: The focus of this section is the configuration of IRB interfaces on Layer 3 VXLAN gateways. For a more comprehensive example of configuring EVPN-VXLAN, see "Example: Configuring EVPN-VXLAN In a Collapsed IP Fabric Topology Within a Data Center" on page 451.

The key to setting up this feature is the configuration of IRB interfaces on Layer 3 VXLAN gateways. In general, you configure the IRB interfaces as you would with only IPv4 hosts in the topology. However, in addition to specifying IPv4 addresses for the IRB interface, you also specify IPv6 addresses. You must also configure a MAC address for the IRB interface.

Table 13 on page 282 shows how to configure the addresses for IRB interfaces irb.100 and irb.200 on the three Layer 3 VXLAN gateways in a one -layer IPv4 fabric.
Table 14 on page 283 outlines some additional required global IRB interface configuration.

Table 13: IPv4 Fabric Scenario 2—Sample IRB Interface Addresses

Addresses	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3	Description
IRB interface irb.100				
IPv4 address	192.168.100.1/24	192.168.100.1/24	192.168.100.1/24	Specify the same IPv4 address for irb.100 on each device.
Global IPv6 address	2001:db8::192.168.100.1/96	2001:db8::192.168.100.1/96	2001:db8::192.168.100.1/96	Specify the same IPv6 address for irb.100 on each device.
Link-local IPv6 address	fe80::100:00:01/96	fe80::100:00:01/96	fe80::100:00:01/96	Specify the same link-local IPv6 address for irb.100 on each device. Any packet destined to this IPv6 address is intercepted for NDP processing.
IRB MAC address	10:00:00:00:00:fe	10:00:00:00:00:fe	10:00:00:00:00:fe	Specify the same MAC address for irb.100 on each device.
IRB interface irb.200				

Table 13: IPv4 Fabric Scenario 2—Sample IRB Interface Addresses (continued)

Addresses	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3	Description
IPv4 address	192.168.200.1/24	192.168.200.1/24	192.168.200.1/24	Specify the same IPv4 address for irb.200 on each device.
Global IPv6 address	2001:db8::192.168.200.1/96	2001:db8::192.168.200.1/96	2001:db8::192.168.200.1/96	Specify the same IPv6 address for irb.200 on each device.
Link-local IPv6 address	fe80::200:00:01/96	fe80::200:00:01/96	fe80::200:00:01/96	Specify the same link-local IPv6 address for irb.200 on each device. Any packet destined to this IPv6 address is intercepted for NDP processing.
IRB MAC address	10:00:00:00:00:ff	10:00:00:00:00:ff	10:00:00:00:00:ff	Specify the same MAC address for irb.200 on each device.

Table 14: IPv4 Fabric Scenario 2—Required IRB Interface Configuration

Description	Layer 2 and 3 VXLAN Gateway 1	Layer 2 and 3 VXLAN Gateway 2	Layer 2 and 3 VXLAN Gateway 3
Global IRB interface configuration			
For IPv4 fabric 2, a default Layer 3 gateway is not configured. Therefore, you must disable the advertisement of a default Layer 3 gateway.	**set protocols evpn default-gateway do-not-advertise**	**set protocols evpn default-gateway do-not-advertise**	**set protocols evpn default-gateway do-not-advertise**

Release History Table

Release	Description
15.1X53-D30	Starting with Junos OS Release 15.1X53-D30 for QFX10000 switches and Junos OS Release 18.3R1 for QFX5110 switches, the IRB interfaces on these Layer 3 VXLAN gateways can route Layer 2 or 3 data packets from one IPv6 host to another IPv6 host through an EVPN-VXLAN network with an IPv4 underlay.

Related Documentation

- Using a Default Layer 3 Gateway to Route Traffic Between Virtual Networks in an EVPN-VXLAN Topology on page 293

CHAPTER 8

Configuring VLAN-Aware Bundle Services, VLAN-Based Services, and Virtual Switch Support

Understanding VLAN-Aware Bundle and VLAN-Based Service for EVPN

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. When deploying the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

The DCSP might require Ethernet VLAN services to be extended over a WAN with a single EVPN instance (EVI). On the QFX Series, each EVPN and virtual switch routing instance corresponds to an EVI. The QFX Series supports VLAN-aware bundle service and VLAN-based service, which maintain data and control plane separation.

NOTE: If you create VLANs that are not part of a routing instance, they become part of the Default Switch routing instance.

VLAN-Aware Bundle Service

VLAN-aware bundle services supports the mapping of one or more routing instances of type Virtual Switch to many VLAN IDs (VIDs) and multiple bridge tables, with each bridge table corresponding to a different VLAN. To enable VLAN-aware bundle service, configure a Virtual Switch routing instance. For service provider-related applications, where the VLAN ID is local to the Layer 2 logic interface, enable the **flexible-vlan-tagging** statement in your configuration. For enterprise-related applications, where the VLAN ID has global significance, enable the **family ethernet-switching** statement in your configuration. VLAN-aware bundle service supports up to 4000 VLANs per routing instance.

VLAN-Based Service

VLAN-based service supports the mapping of one routing instances of type EVPN to one VLAN. There is only one bridge table that corresponds to the one VLAN. If the VLAN consists of multiple VLAN IDs (VIDs)—for example, there is a different VID per Ethernet segment on a provider edge device—then VLAN translation is required for packets that are destined to the Ethernet segment. The Ethernet tag ID in To enable VLAN-based service, configure an EVPN routing instance. Up to 100 EVPN routing instances are supported.

Configuring EVPN with VLAN-Based Service

VLAN-based service supports the mapping of one or more routing instances of type EVPN to only one VLAN. There is only one bridge table that corresponds to the one VLAN. If the VLAN consists of multiple VLAN IDs (VIDs)—for example, there is a different VID per Ethernet segment on a provider edge device—then VLAN translation is required for packets that are destined to the Ethernet segment.

To configure VLAN-based service and Layer 3 routing with two EVPN routing instances on a provider edge device.

1. Configure the first routing instance of type **evpn** named **evpn1**.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 instance-type evpn
   ```

2. Configure the access interface for handling EVPN traffic.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 interface xe-0/0/8.100
   ```

3. Configure a Layer 3 integrated and routing (IRB) interface for the **evpn1** routing instance.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 l3-interface irb.100
   ```

4. Configure the Virtual Tunnel Endpoint interface for the **evpn1** routing instance.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 vtep-source-interface lo0.0
   ```

5. Configure a VLAN identifier for the **evpn1** routing instance.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 vlan-id none
   ```

6. Configure a route distinguisher for the **evpn1** routing instance.

   For example:

   ```
   [edit]
   user@switch# routing-instances evpn1 route-distinguisher 1.2.3.11:1
   ```

7. Configure the VPN routing and forwarding (VRF) target community for the **evpn1** routing instance.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 vrf-target target:1234:11
   ```

8. Configure the encapsulation type for the **evpn1** routing instance.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 protocols evpn encapsulation vxlan
   ```

9. Configure the VXLAN Network Identifier (VNI) for the **evpn1** routing instance.

   For example:

   ```
   [edit]
   user@switch# set routing-instances evpn1 vxlan vni 100
   ```

10. Configure the second routing instance of type **evpn** named **evpn2**:

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 instance-type evpn
    ```

11. Configure the access interface on the provider edge device (PE) for handling EVPN traffic.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 interface xe-0/0/8.200
    ```


12. Configure a Layer 3 integrated and routing (IRB) interface for the **evpn2** routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 l3-interface irb.200
    ```

13. Configure the loopback address as the virtual tunnel endpoint source interface for the **evpn2** routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 vtep-source-interface lo0.0
    ```

14. Configure the VLAN identifier for the **evpn2** routing instance to be **none**.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 vlan-id none
    ```

15. Configure a route distinguisher for the **evpn2** routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 route-distinguisher 1.2.3.11:2
    ```

16. Configure the VPN routing and forwarding (VRF) target community for the **evpn2** routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 vrf-target target:1234:24
    ```

17. Configure the encapsulation type for the **evpn2** routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 protocols evpn encapsulation vxlan
    ```

18. Configure the VXLAN Network Identifier (VNI) for the **evpn2** routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances evpn2 vxlan vni 200
    ```

19. Configure a VPN routing and forwarding (VRF) routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances vrf instance-type vrf
    ```

20. Configure the first of two integrated routing and bridging (IRB) interface for the **vrf** instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances vrf interface irb.100
    ```

21. Configure the second of two integrated routing and bridging (IRB) interface for the VPN routing and forwarding (VRF) instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances vrf interface irb.200
    ```

22. Configure the loopback interface for the **vrf** instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances vrf interface lo0.1000
    ```

23. Configure a unique route distinguisher for the VPN routing and forwarding (VRF) instance to identify from which EVPN the route belongs.

    For example:

    ```
    [edit]
    user@switch# set routing-instances vrf route-distinguisher 1.2.3.1:2
    ```

24. Configure the VPN routing and forwarding (VRF) target community for the VPN routing and forwarding (VRF) routing instance.

    For example:

    ```
    [edit]
    user@switch# set routing-instances vrf vrf-target target:2222:22
    ```

25. Configure the **auto-export** option to automatically derive the route target.

    For example:

    ```
    [edit]
    user@switch# set routing-instances vrf routing-options auto-export
    ```

Virtual Switch Support for EVPN Overview

Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers. Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches. Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches. . This feature allows Ethernet VLANs over a WAN to share a single EVPN instance while maintaining data-plane separation between the different VLANs.

Junos OS has a highly flexible and scalable virtual switch interface. With a virtual switch, a single router or switch can be divided into multiple logical switches. Layer 2 domains (also called *bridge-domains* or *vlans*) can be defined independently in each virtual switch. To configure VLAN-aware bundle service, an EVPN must run in a virtual-switch routing instance.

On the EX Series and MX Series, a single EVPN instance can stretch up to 4094 bridge domains or VLANs defined in a virtual switch to remote sites. A virtual switch can have more than 4094 bridge domains or VLANs with a combination of none, single, and dual VLANs. However, because EVPN signaling deals only with single VLAN tags, a maximum of 4094 bridge domains or VLANs can be stretched. The EVPN virtual switch also provides support for trunk and access interfaces.

On the QFX10000 switches, Layer 2 VLANs use the default-switch routing instance. An EVPN instance (EVI) is not supported.

NOTE:

- The none VLAN option is supported with bridge domains or VLANs under the virtual switch instance type for EVPNs.
- Access interfaces configured with single or dual VLAN tags are supported in EVPN. By default, only Ethernet frames with single or no VLAN tags are transported across the EVPN core. As a result, dual-tagged Ethernet frames received on the access interfaces must be normalized to Ethernet frames with single or no VLAN tags for proper transmission over the EVPN core.

  You can enable transporting of dual-tagged frames across the EVPN core network by including both the **vlan-id none** and **no-normalization** configuration statements together.

There are two types of VLAN-aware bundle service:

- VLAN-aware bundle without translation

  The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance with a guarantee end-to-end customer VLAN transparency. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN.

  NOTE: The QFX10000 switches only support VLAN-aware bundle service interface without translation. Support is limited to 4K VLANs because only the default-switching instance is supported.

- VLAN-aware bundle with translation

  The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN. The service interface supports

customer VLAN translation to handle the scenario where different VLAN Identifiers (VIDs) are used on different interfaces to designate the same customer VLAN.

EVPN with virtual switch provides support for VLAN-aware bundle with translation only.

Release History Table

Release	Description
17.3	Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches.
14.2	Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches.
14.1	Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers.

Related Documentation

- Example: Configuring EVPN with Support for Virtual Switch on page 587

Configuring EVPN with Support for Virtual Switch

You can configure an Ethernet VPN (EVPN) with virtual switch support to enable multiple tenants with independent VLAN and subnet space within an EVPN instance. Virtual switch provides the ability to extend Ethernet VLANs over a WAN using a single EVPN instance while maintaining data-plane separation between the various VLANs associated with that instance. A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites.

When configuring virtual switch for EVPN, be aware of the following considerations:

- Due to default ARP policing, some of the ARP packets not destined for the device can be missed. This can lead to delayed ARP learning and synchronization.
- Clearing ARP for an EVPN can lead to inconsistency between the ARP table and the EVPN ARP table. To avoid this situation, clear both ARP and EVPN ARP tables.
- The **vlan-tag** can be configured for local switching. However, vlan-tagged VLANs should not be extended over the EVPN cloud.

This task explains how to configure one Virtual Switch instance that includes one VLAN.

1. Configure the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type virtual-switch
```

2. Configure the interface names for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface interface-name
```

3. Configure the route distinguisher for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

4. Configure the VPN routing and forwarding (VRF) target community for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target
```

5. List the VLAN identifiers that are to be EVPN extended.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn extended-vni-list [vlan-id-range]
```

6. Configure the VLAN and VLAN ID for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vlans name of VLAN vlan-id VLAN ID number
```

7. Configure VXLAN encapsulation and Virtual Network Identifier for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vxlan vni VNI number
```

8. Configure the virtual tunnel endpoint source interface for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vtep-source-interface interface-name
```

9. Verify and commit the configuration.

```
[edit]
user@PE1# commit
commit complete
```

Related Documentation

- Example: Configuring EVPN with Support for Virtual Switch on page 587

CHAPTER 9

Setting Up a Layer 3 VXLAN Gateway

Using a Default Layer 3 Gateway to Route Traffic Between Virtual Networks in an EVPN-VXLAN Topology

Physical (bare-metal) servers in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment rely on a default Layer 3 gateway to route their traffic from one virtual network (VN) to another physical server or a virtual machine (VM) in another VN. You can enable the default gateway functionality on a Juniper Networks device that acts as a Layer 3 VXLAN gateway. On a Layer 3 VXLAN gateway, you can configure an integrated routing and bridging (IRB) interface with a virtual gateway address (VGA), which in turn configures the IRB interface as a default Layer 3 gateway. You can configure an IRB interface with a VGA when using EVPN-VXLAN within a data center and across the Data Center Interconnect (DCI) solution.

This topic covers the following:

Understanding the Default Gateway

To enable the default gateway function, you configure an IRB interface with a unique IP address and a media access control (MAC) address. In addition, you configure the IRB interface with a VGA, which must be an anycast IP address, and the Layer 3 VXLAN gateway automatically generates a MAC address.

When you specify an IPv4 address for the VGA, the Layer 3 VXLAN gateway automatically generates 00:00:5e:00:01:01 as the MAC address. When you specify an IPv6 address, the Layer 3 VXLAN gateway automatically generates 00:00:5e:00:02:01 as the MAC address.

NOTE: On Juniper Networks devices that function as Layer 3 VXLAN gateways, you can explicitly configure an IPv4 or IPv6 MAC address for a default gateway by using the **virtual-gateway-v4-mac** or **virtual-gateway-v6-mac** configuration statement at the **[edit interfaces name irb unit logical-unit-number]** hierarchy level. After you perform this configuration, the automatically generated MAC address is overridden by the configured MAC address.

A VGA and associated MAC address provide the default gateway function in a particular VN. You configure each host (physical server or VM) in the VN to use the VGA.

By using an anycast IP address as the VGA, when a VM is moved from one EVPN provider edge (PE) device to another in the same VN, the VM can use the same default gateway. In other words, you do not need to update the VM with a new default gateway IP address for MAC binding.

Layer 3 VXLAN gateways in an EVPN-VXLAN topology respond to Address Resolution Protocol (ARP) requests for the VGA and forward packets intended for the default gateway MAC address.

BEST PRACTICE: On Juniper Networks devices that function as Layer 3 VXLAN gateways in an EVPN-VXLAN topology with a two-layer IP fabric, we recommend that you configure each IRB interface with unique IP and MAC addresses. We issue this recommendation to avoid an asymmetric data path for the ARP request and response when the IRB interface sends ARP messages intended for an end-destination's MAC address.

The exception to this recommendation is the QFX5110 switch, which does not support the configuration of an IRB interface with a unique MAC address.

For IRB interfaces configured on QFX10000 switches in an EVPN-VXLAN topology with a collapsed IP fabric, you can alternatively configure each IRB interface on each Layer 3 VXLAN gateway in a VN with the same MAC address. For more information, see "Example: Configuring EVPN-VXLAN In a Collapsed IP Fabric Topology Within a Data Center" on page 451.

NOTE: To troubleshoot an IRB interface, you can ping the IP address of the interface.

To troubleshoot a default gateway on an MX Series router, you can ping the VGA of the default gateway from a CE device. To support pinging of the VGA, include the **virtual-gateway-accept-data** statement at the **[edit interfaces irb unit]** hierarchy of the preferred virtual gateway.

Additionally, you can ping the IP address of the CE device from the PE device (MX Series router). To support pinging of the IP address of the CE device, include the preferred statement at **[edit interfaces irb unit *logical-unit-number* family (inet |inet6} address *ip-address*]** hierarchy using the unique IRB IP address. Otherwise , you must manually specify the unique IRB IP address as the source IP address when you ping the CE device.

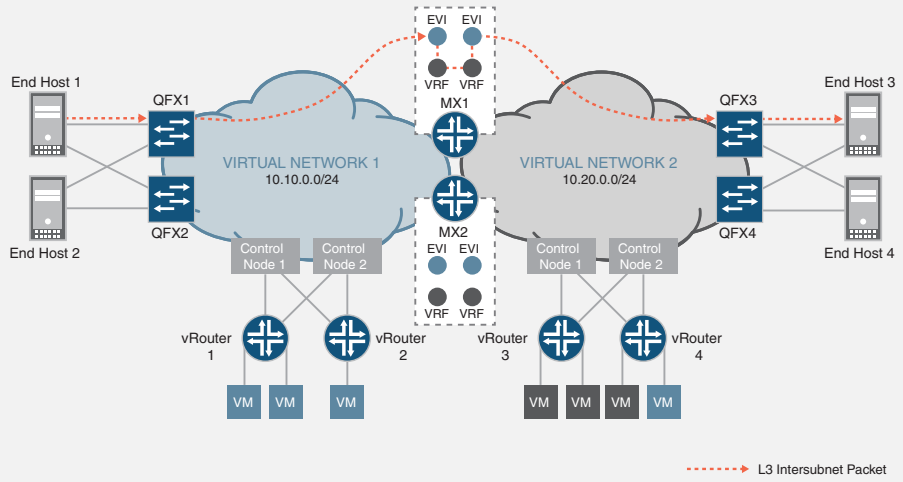
For each IRB interface with a VGA configured, there are two sets of IP and MAC addresses—one set for the IRB interface itself and one set for the default gateway. As a result, MAC routes for both IRB interface and default gateway are advertised. However, no default gateway extended community attribute is associated with the MAC route advertisement for the default gateway because all Layer 3 VXLAN gateways have the same anycast IP address and MAC binding.

- Understanding How a Default Gateway Handles Known Unicast Traffic Between Virtual Networks on page 295
- Understanding How a Default Gateway Handles Unknown Unicast Traffic Between Virtual Networks on page 296

Understanding How a Default Gateway Handles Known Unicast Traffic Between Virtual Networks

In the EVPN-VXLAN topology with a two-layer IP fabric shown in Figure 25 on page 296, MX Series routers function as Layer 3 VXLAN gateways and QFX5200 switches function as Layer 2 VXLAN gateways. End hosts 1 through 4 are physical servers that must communicate with each other.

Figure 25: Handling Known Unicast Traffic Between Virtual Networks



In this topology, end host 1 in VN1 (10.10.0.0/24) and end host 3 in VN 2 (10.20.0.0/24) exchange known unicast packets. Before the exchange of packets between the two end hosts, assume that the hosts sent ARP requests to MX1, which is a Layer 3 VXLAN gateway, and that MX1 responded with the MAC address of a default gateway in VN1.

For example, end host 1 originates a packet and sends it to QFX1, which is a Layer 2 VXLAN gateway. QFX1 encapsulates the packet with a VXLAN header and sends it to MX1. For the inner destination MAC, the packet includes the MAC address of a default gateway in VN1. For the inner destination IP, the packet includes the IP address of end host 3. Upon receipt of the packet, MX1 de-encapsulates it, and after detecting the MAC address of the default gateway in the inner destination MAC field, performs a route lookup for end host 3's IP address in the L3-VRF routing table for VN1. After a route is found, the packet is routed to VN2 and based on the ARP route entry, the packet is encapsulated with a VXLAN header and sent to QFX3. QFX3 de-encapsulates the packet, and sends it to end host 3.

NOTE: The traffic flow and handling of known unicast traffic in an EVPN-VXLAN topology with a collapsed IP fabric are essentially the same as described in this section. The only difference is that in the collapsed topology, a QFX Series switch that supports Layer 3 VXLAN gateway functionality acts as both Layer 2 and Layer 3 VXLAN gateways.

Understanding How a Default Gateway Handles Unknown Unicast Traffic Between Virtual Networks

NOTE: The information in this section applies to the traffic flow and handling of unknown unicast packets in EVPN-VXLAN topologies with both two-layer and collapsed IP fabrics.

For unknown unicast traffic between VNs that is initiated by a physical server, an additional ARP request and response process is required at each stage. After the destination MAC addresses for both default gateway and host is resolved, the traffic flows in the same way as described in "Understanding How a Default Gateway Handles Known Unicast Traffic Between Virtual Networks" on page 295.

Understanding the Redundant Default Gateway

The Juniper Networks devices that function as Layer 3 VXLAN gateways can also provide redundant default gateway functionality. A redundant default gateway prevents the loss of communication between physical servers in one VN and physical servers or VMs in another VN.

The redundant default gateway functionality is typically achieved in an EVPN-VXLAN topology where a provider edge (PE) device such as a Layer 2 VXLAN gateway or a Contrail vRouter is multihomed in active-active mode to multiple Layer 3 VXLAN gateways. On the Layer 3 VXLAN gateways, IRB interfaces are configured as default gateways. Note that each default gateway uses the same VGA and MAC address. In addition, the VGAs and MAC addresses are associated with the same Ethernet segment ID (ESI).

The ESI associated with the VGA and MAC address of the default gateway is automatically derived from an autonomous system (AS) and the VXLAN network identifier (VNI) for the VN. As a result, the default gateway MAC routes advertised by each Layer 3 VXLAN gateway for a given VN have the same ESI.

From the perspective of a Layer 2 VXLAN gateway or a Contrail vRouter that is multihomed to the Layer 3 VXLAN gateways, the addresses of each default gateway configured on each Layer 3 VXLAN gateway is the same. As a result, the PE devices build an equal-cost multipath (ECMP) next hop to reach each default gateway. Traffic that originates from a host and is destined for the MAC address of a default gateway is load balanced.

If one of the Layer 3 VXLAN gateways fails, the remote PE devices are notified of the withdrawing or purging of the next hop to the default gateway MAC address. The path to the failed Layer 3 VXLAN gateway is removed from the next-hop database. Despite the removal of the path, the default gateway that is configured on the remaining Layer 3 VXLAN gateway is still reachable, and the ARP entries for the hosts remain unchanged.

Understanding Dynamic ARP Processing

When a physical server needs to determine the MAC address of its default gateway, the physical server initiates an ARP request that includes the VGA of the default gateway. In an EVPN-VXLAN topology with a two-layer IP fabric, a Layer 2 VXLAN gateway typically receives the ARP request, encapsulates the request in a VXLAN header, and forwards the encapsulated packet to a Layer 3 VXLAN gateway. In an EVPN-VXLAN topology with a collapsed IP fabric, a Layer 2 and 3 VXLAN gateway typically receives the ARP request from the directly connected physical server.

Upon receipt of the ARP request, the Layer 3 VXLAN gateway de-encapsulates the packet if appropriate, learns the IP and MAC binding of the physical server, and creates an ARP entry in its database. The Layer 3 VXLAN gateway then replies with the MAC address of the default gateway.

In an EVPN-VXLAN topology with a two-layer IP fabric, the ARP response is encapsulated with a VXLAN header and unicast back to the Layer 2 VXLAN gateway. The Layer 2 VXLAN gateway de-encapsulates the ARP response and forward the packet to the physical server.

In an EVPN-VXLAN topology with a collapsed IP fabric, the ARP response is unicast back to the directly connected physical server.

In a situation where a physical server in VN1 originates a packet that is destined for a physical server in VN2, the Layer 3 VXLAN gateway searches its database for an ARP entry for the destination physical server. If a match is not found, the Layer 3 VXLAN gateway initiates an ARP request that includes the IP and MAC addresses of the IRB interface that is mapped to VN2, and sends the request to the destination physical server. The destination physical server learns the IP/MAC binding of the IRB interface, and adds or refreshes the ARP entry in its database accordingly. The physical server then unicasts an ARP response, which includes the MAC address of the IRB interface, back to the Layer 3 VXLAN gateway,

Related Documentation

- EVPN Multihoming Overview on page 29
- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches on page 270
- EVPN-over-VXLAN Supported Functionality on page 255

Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Topology

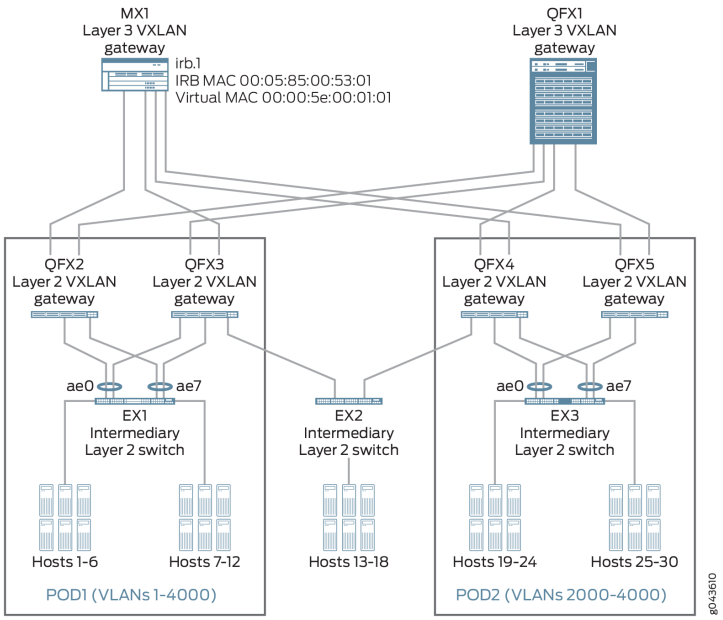
In an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) topology with a two-layer IP fabric, an MX Series router or a QFX10000 switch can function as a Layer 3 VXLAN gateway on which you can configure integrated routing and bridging (IRB) interfaces. The configuration of each IRB interface can also include a virtual gateway address (VGA), which creates a default Layer 3 virtual gateway with the specified IP address. Through the IRB interface with which it is configured, the default virtual gateway enables the communication between non-virtualized hosts, virtual machines (VMs), and servers in different VXLANs or IP subnetworks.

When you configure a VGA for an IRB interface, the Layer 3 VXLAN gateway automatically generates IPv4 media access control (MAC) address 00:00:5E:00:01:01 or IPV6 MAC address 00:00:5E:00:02:01 for that particular virtual gateway. (This topic refers to the virtual gateway MAC address as a *virtual MAC*.) The automatically generated virtual MAC is not included as the source MAC address in packets generated by the Layer 3 VXLAN gateway. Instead, data packets and the source MAC address field in the outer Ethernet header of Address Resolution Protocol (ARP) replies and neighbor advertisement packets include the MAC address for the IRB interface. (This topic refers to the MAC address for the IRB interface as the *IRB MAC*.)

When an ARP reply includes the IRB MAC as the source MAC address instead of the virtual MAC, an issue might arise in an EVPN-VXLAN topology with a two-layer IP fabric.

For example, in the topology shown in Figure 26 on page 299, an MX Series router and a QFX10000 switch function as Layer 3 VXLAN gateways, and four QFX5100 switches function as Layer 2 VXLAN gateways. Also included in the topology are three intermediary Layer 2 switches, in this case, EX4300 switches, to which hosts are connected.

Figure 26: EVPN-VXLAN Topology With Two-Layer IP Fabric



On the MX Series router, an IRB interface named irb.1 has a MAC address of 00:05:85:00:53:01 and a VGA of 10.2.1.254. The MX Series router automatically generates the MAC address 00:00:5e:00:01:01 for the default virtual gateway.

In this topology, irb.1 on the MX Series router receives an ARP request from host 1. In its ARP reply, the MX Series router includes the following:

- Source MAC address in outer Ethernet header: 00:05:85:00:53:01 (IRB MAC) → intermediary Layer 2 switch EX1 learns this MAC address.
- Sender MAC address within ARP reply packet: 00:00:5e:00:01:01 (virtual MAC) → intermediary Layer 2 switch EX1 cannot see this MAC address, and therefore, does not learn it.

When intermediary Layer 2 switch EX1 receives the ARP reply, it learns only the source MAC address (IRB MAC). As a result, if host 1 sends packets that include the virtual MAC in the header, EX1 is unable to find the virtual MAC in its MAC table. Therefore, EX1 floods the domain with unknown-unicast packets.

NOTE: The flooding of unknown-unicast packets is not an issue in EVPN-VXLAN topologies with a collapsed IP fabric. That is, a topology in which a single layer of QFX10000 switches function as both Layer 3 and Layer 2 VXLAN gateways. In the collapsed IP fabric topology, hosts are directly connected to the Layer 3 and Layer 2 VXLAN gateways. Further, each IRB interface is typically configured with an IP address and a static MAC address. The configuration of each IRB interface on a particular VXLAN gateway is repeated on each gateway in the collapsed IP fabric. With the same MAC address configured for each IRB interface on each VXLAN gateway, each host uses the same MAC address when sending inter-VXLAN traffic regardless of where the host is located or which VXLAN gateway receives the traffic. These factors make the configuration of a default virtual gateway unnecessary. For more information about the collapsed IP fabric topology, see "Example: Configuring EVPN-VXLAN In a Collapsed IP Fabric Topology Within a Data Center" on page 451.

Starting with Junos OS Release 14.2R5 for MX Series routers and Junos OS Release 15.1X53-D63 for QFX10000 switches, you can explicitly configure an IPv4 or IPv6 MAC address for a default virtual gateway by using the virtual-gateway-v4-mac or virtual-gateway-v6-mac configuration statement at the **[edit interfaces *name* irb unit *logical-unit-number*]** hierarchy level. After you perform this configuration, the automatically generated virtual MAC is overridden by the configured virtual MAC. That is, when Layer 3 VXLAN gateway MX1 sends data packets, ARP replies, and neighbor advertisement packets, the configured virtual MAC is in the outer Ethernet header of these packets. As a result, interrmediary Layer 2 switch EX1 also learns the configured virtual MAC, thereby eliminating the possibility that the switch floods the domain with unknown-unicast packets.

Release History Table

Release	Description
14.2R5	Starting with Junos OS Release 14.2R5 for MX Series routers and Junos OS Release 15.1X53-D63 for QFX10000 switches, you can explicitly configure an IPv4 or IPv6 MAC address for a default virtual gateway by using the virtual-gateway-v4-mac or virtual-gateway-v6-mac configuration statement at the **[edit interfaces *name* irb unit *logical-unit-number*]** hierarchy level.

Supported Protocols on an IRB Interface in EVPN-VXLAN

EVPN-VXLAN provides logical layer 2 connectivity over a layer 3 network. It is a logical overlay that is decoupled from the underlay network. The layer 2 subnets in the EVPN-VXLAN network are uniquely identified by the virtual network identifier (VNI) where devices configured with the same VNI are considered to be in the same logical subnet. The devices in the same logical subnet can communicate directly with each other when they are connected to the same virtural gateway. To route traffic when devices with different VNIs are connected to the same gateway,, you must configure an IRB interface on the VXLAN gateway to route traffic. This allows protocol adjacencies to be established between the layer 3 gateway IRB interface and customer edge devices in the following scenarios:

- Support networking functions when there are firewalls or load balancers connected between the host and the gateway device.
- Support inter-VNI routing when VNIs for the same tenant extend across data centers.

Figure 27 on page 301 illustrates a simple leaf-spine topology in an EVPN-VXLAN network with the VXLAN gateway on spine devices. For host 1 to communicate with hosts in other VNIs, you must configure the IRB interfaces on the VXLAN gateway on the spine devices .

Figure 27: Leaf-spine Topology in an EVPN-VXLAN network.

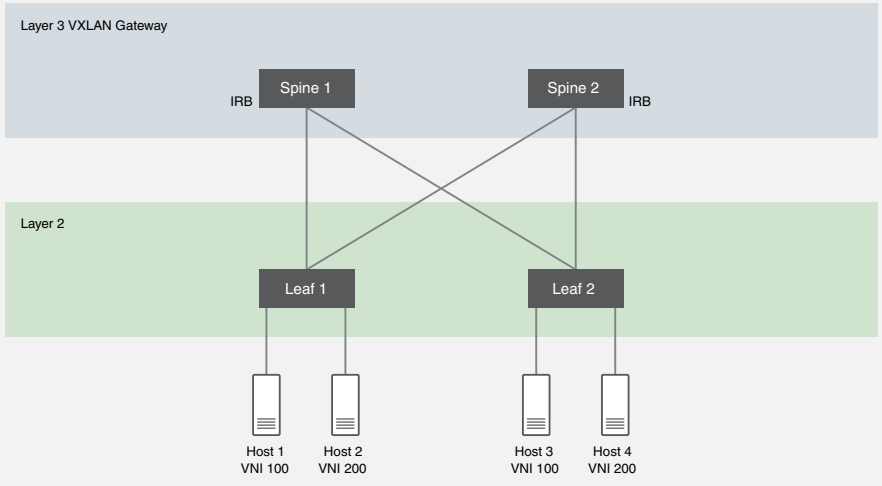
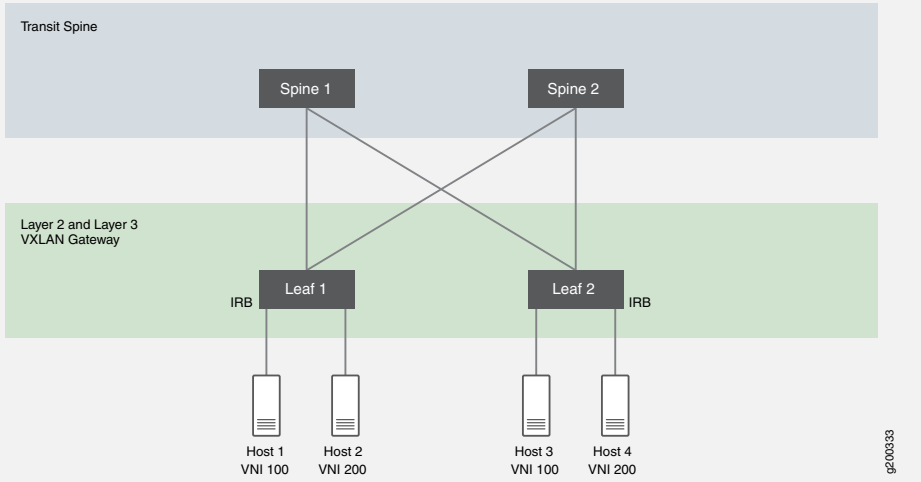


Figure 28 on page 302 illustrates a collapsed leaf-spine topology with VXLAN gateways on leaf devices. For host 1 to communicate with hosts in other VNIs, you must configure IRB interfaces on the VXLAN gateway on the leaf devices.

Figure 28: Collapsed Leaf-spine Topology in an EVPN-VXLAN network.



Junos OS supports the following protocols on the IRB in the EVPN-VXLAN overlay network:

- OSPF
- IS-IS
- BGP (eBGP and iBGP)
- Bidirectional forwarding detection (BFD) support for ISIS, OSPF, BGP and static routing.

Related Documentation

- *Understanding OSPF Routing Instances*
- *Understanding IS-IS Configuration*
- *Understanding External BGP Peering Sessions*
- *Understanding Internal BGP Peering Sessions*
- *Understanding BFD for Static Routes for Faster Network Failure Detection*

Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center

Starting with Junos OS Release 15.1X53-D30, you can use integrated routing and bridging (IRB) interfaces to route data packets between Virtual Extensible LANs (VXLANs) that are implemented in an Ethernet VPN (EVPN)-VXLAN environment. In this example, the

IRB interfaces provide Layer 3 connectivity for physical servers, on which applications directly run, in the same data center.

Requirements

This example uses the following hardware and software components:

- Two spine devices:
  - A QFX10008 switch (Spine 1) running Junos OS Release 15.1X53-D30 software.
  - A QFX10002 switch (Spine 2) running Junos OS Release 15.1X53-D30 software.
- Four QFX5100 switches (Leaf 1 through Leaf 4) running Junos OS Release 14.1X53-D30 software.
- Leaf 1 and Leaf 2 that are connected to one physical server, and Leaf 3 and Leaf 4 that are connected to another physical server.

Overview

In this example, physical servers that support two groups in a customer's organization must exchange large volumes of data. To meet this need, the following protocols are implemented:

- EVPN is used to establish a Layer 2 virtual bridge to connect the two physical servers.
- Within the EVPN topology, BGP is used to exchange route information.
- VXLAN is used to tunnel the data packets through the underlying Layer 3 fabric. The use of VXLAN encapsulation preserves the Layer 3 fabric for use by routing protocols.
- IRB interfaces are used to route data packets between the VXLANs.

In this example, IRB interfaces are configured on the spine devices only. With this configuration, the spine devices function as Layer 3 gateways for the physical servers that are connected through their respective leaf devices. When the physical servers in the two customer sites exchange data, the spine devices route the data packets to their respective destinations.

Topology

The simple IP Clos topology shown in Figure 29 on page 304 includes two spine switches, four leaf switches, and two physical servers. Each leaf switch has a connection to each of the spine switches for redundancy. Physical server A supports group 1, and physical server B supports group 2 in the same customer organization. The customer network is segmented into four VXLANs. VXLANs v0001 and v0002 support group 1, and VXLANs v0005 and v0006 support group 2. Table 15 on page 304 shows key entities, including IRB interfaces, that are configured for customer groups 1 and 2. Each VXLAN is supported by an IRB interface, over which data packets from the other VXLANs are routed.

Figure 29: EVPN-VXLAN Topology with IRB Interfaces

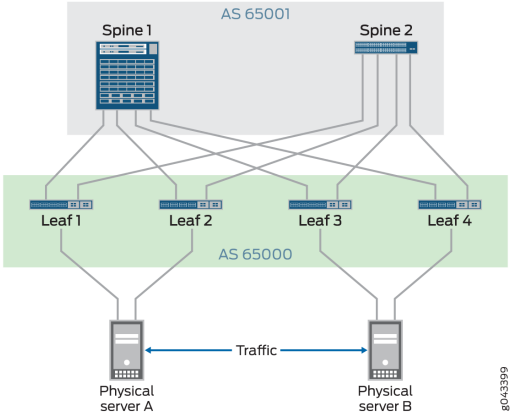


Table 15: Key Entities Configured for Customer Groups 1 and 2

Entity	Entity for Customer Group 1	Entity for Customer Group 2
VXLAN	v0001	v0005
	v0002	v0006

Table 15: Key Entities Configured for Customer Groups 1 and 2 (continued)

Entity	Entity for Customer Group 1	Entity for Customer Group 2
VNI	1	5
	2	6
IRB interface	irb.1	irb.5
	irb.2	irb.6

When configuring the entities in Table 15 on page 304, keep the following in mind:

- Each VXLAN must be assigned an IP address with a different IP subnet than the other VXLANs.
- Each VXLAN must be assigned a unique VXLAN network identifier (VNI).
- Each IRB interface must be specified as part of a Layer 3 virtual routing fowarding (VRF) instance.
- The configuration of each IRB interface must include a default gateway address, which you can specify by including the **virtual-gateway-address** configuration statement at the **[interfaces irb unit *logical-unit-number* family inet address *ip-address*]** hierarchy level. Configuring the default gateway sets up a redundant default gateway for each IRB interface.
- Policies must be set up to enable the exchange of data packets between the four VXLANs.

NOTE: When setting up a routing instance for EVPN-VXLAN, you must include a loopback interface. You must also specify an IP address for the interface using the **set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix***. If you inadvertently omit the loopback interface and associated IP address, EVPN control packets cannot be processed.

Spine 1: Underlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces et-1/0/0 vlan-tagging
set interfaces et-1/0/0 unit 0 vlan-id 1
set interfaces et-1/0/0 unit 0 family inet address 10.1.24.10/24
set interfaces et-1/0/1 unit 0 family inet address 10.1.25.10/24
set interfaces et-1/0/4 unit 0 family inet address 10.1.28.10/24
set interfaces et-1/0/5 unit 0 family inet address 10.1.29.10/24
```

```
set interfaces et-2/0/0 vlan-tagging
set interfaces et-2/0/0 unit 0 vlan-id 1
set interfaces et-2/0/0 unit 0 family inet address 10.1.30.10/24
set interfaces et-2/0/1 unit 0 family inet address 10.1.31.10/24
set interfaces et-2/0/4 unit 0 family inet address 10.1.34.10/24
set interfaces et-2/0/5 unit 0 family inet address 10.1.35.10/24
set interfaces lo0 unit 0 family inet address 10.0.0.9/32 primary
set routing-options router-id 10.0.0.9
set routing-options autonomous-system 65001
set routing-options forwarding-table export load-balancing-policy
set protocols bgp multihop
set protocols bgp group underlay type external
set protocols bgp group underlay accept-remote-nexthop
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 5
set protocols bgp group underlay export send-direct
set protocols bgp group underlay peer-as 65000
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.1.24.3
set protocols bgp group underlay neighbor 10.1.25.4
set protocols bgp group underlay neighbor 10.1.28.7
set protocols bgp group underlay neighbor 10.1.29.8
set protocols bgp group underlay neighbor 10.1.30.3
set protocols bgp group underlay neighbor 10.1.31.4
set protocols bgp group underlay neighbor 10.1.34.7
set protocols bgp group underlay neighbor 10.1.35.8
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 10 from protocol direct
set policy-options policy-statement send-direct term 10 then accept
```

Spine 1: Configure the Underlay Network

Step-by-Step Procedure

To configure the underlay network on Spine 1:

1. Configure Layer 2 and Layer 3 interfaces.

```
[edit interfaces]
user@switch# set et-1/0/0 vlan-tagging
user@switch# set et-1/0/0 unit 0 vlan-id 1
user@switch# set et-1/0/0 unit 0 family inet address 10.1.24.10/24
user@switch# set et-1/0/1 unit 0 family inet address 10.1.25.10/24
user@switch# set et-1/0/4 unit 0 family inet address 10.1.28.10/24
user@switch# set et-1/0/5 unit 0 family inet address 10.1.29.10/24
user@switch# set et-2/0/0 vlan-tagging
user@switch# set et-2/0/0 unit 0 vlan-id 1
user@switch# set et-2/0/0 unit 0 family inet address 10.1.30.10/24
user@switch# set et-2/0/1 unit 0 family inet address 10.1.31.10/24
user@switch# set et-2/0/4 unit 0 family inet address 10.1.34.10/24
user@switch# set et-2/0/5 unit 0 family inet address 10.1.35.10/24
```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.0.0.9/32 primary
```

3. Set the routing options.

```
[edit routing-options]
user@switch# set router-id 10.0.0.10
user@switch# set autonomous-system 65001
user@switch# set forwarding-table export load-balancing-policy
```

4. Configure multihop external BGP (EBGP).

```
[edit protocols]
user@switch# set bgp multihop
```

5. Configure a BGP group that includes peers that handle underlay functions.

```
[edit protocols]
user@switch# set bgp group underlay type external
user@switch# set bgp group underlay accept-remote-nexthop
user@switch# set bgp group underlay advertise-peer-as
user@switch# set bgp group underlay family inet unicast loops 5
user@switch# set bgp group underlay export send-direct
user@switch# set bgp group underlay peer-as 65000
user@switch# set bgp group underlay multipath
user@switch# set bgp group underlay neighbor 10.1.24.3
user@switch# set bgp group underlay neighbor 10.1.25.4
user@switch# set bgp group underlay neighbor 10.1.28.7
user@switch# set bgp group underlay neighbor 10.1.29.8
user@switch# set bgp group underlay neighbor 10.1.30.3
user@switch# set bgp group underlay neighbor 10.1.31.4
user@switch# set bgp group underlay neighbor 10.1.34.7
user@switch# set bgp group underlay neighbor 10.1.35.8
```

NOTE: You must enter the **set protocols bgp group underlay accept-remote-nexthop** command so that the BGP routes appear in the routing table as direct routes. VXLAN requires direct next hops for equal-cost multipath (ECMP) load balancing.

6. Configure a per-packet load-balancing policy.

```
[edit policy-options]
```

```
user@switch# set policy-statement load-balancing-policy then load-balance
  per-packet
```

7. Configure a policy that enables BGP to advertise direct interfaces such as loopback interface lo0.

```
[edit policy-options]
user@switch# set policy-statement send-direct term 10 from protocol direct
user@switch# set policy-statement send-direct term 10 then accept
```

Spine 1: Overlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.0.0.10
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.0.0.7
set protocols bgp group evpn neighbor 10.0.0.8
set protocols bgp group evpn neighbor 10.0.0.9
set protocols bgp group evpn neighbor 10.0.0.4
set protocols bgp group evpn neighbor 10.0.0.3
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
set policy-options community switch_options_comm members target:65000:2
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.0.0.10:1
set switch-options vrf-import EVPN_VRF_IMPORT
set switch-options vrf-target target:65000:2
set switch-options vrf-target auto
```

Configuring the Overlay Network on Spine 1

Step-by-Step Procedure

To configure the overlay network on Spine 1:

1. Configure an internal BGP (iBGP) group for the EVPN-VXLAN overlay.

```
[edit protocols]
user@switch# set bgp group evpn type internal
user@switch# set bgp group evpn local-address 10.0.0.10
```

```
user@switch# set bgp group evpn family evpn signaling
user@switch# set bgp group evpn local-as 65000
user@switch# set bgp group evpn multipath
user@switch# set bgp group evpn neighbor 10.0.0.7
user@switch# set bgp group evpn neighbor 10.0.0.8
user@switch# set bgp group evpn neighbor 10.0.0.9
user@switch# set bgp group evpn neighbor 10.0.0.4
user@switch# set bgp group evpn neighbor 10.0.0.3
```

2. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors.

```
[edit protocols]
user@switch# set evpn encapsulation vxlan
```

3. Specify a tunnel type used for passing multicast traffic between Juniper Networks switches in an EVPN-VXLAN environment.

```
[edit protocols]
user@switch# set evpn multicast-mode ingress-replication
```

4. Set up a policy to handle Layer 2 EVPN traffic.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
   from community switch_options_comm
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
   then accept
user@switch# set community switch_options_comm members target:65000:2
```

5. Configure options for the default routing instance (virtual switch type).

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 10.0.0.10:1
user@switch# set vrf-import EVPN_VRF_IMPORT
user@switch# set vrf-target target:65000:2
user@switch# set vrf-target auto
```

Spine 1: Customer Profile Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

NOTE: When setting up a routing instance for EVPN-VXLAN, you must include a loopback interface. You must also specify an IP address for the interface using the set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix*. If you inadvertently omit the loopback interface and associated IP address, EVPN control packets cannot be processed.

```
set interfaces irb unit 1 family inet address 10.2.1.10/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.10/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 5 family inet address 10.2.5.10/24 virtual-gateway-address 10.2.5.254
set interfaces irb unit 6 family inet address 10.2.6.10/24 virtual-gateway-address 10.2.6.254
set interfaces lo0 unit 1 family inet address 127.10.0.1/32
set interfaces lo0 unit 2 family inet address 127.10.0.2/32
set protocols evpn extended-vni-list [ 1 2 5 6]
set protocols evpn vni-options vni 1 vrf-target export target:10003:1
set protocols evpn vni-options vni 2 vrf-target export target:10003:2
set protocols evpn vni-options vni 5 vrf-target export target:10003:5
set protocols evpn vni-options vni 6 vrf-target export target:10003:6
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 from community
  cust0001
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 from community
  cust0002
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 from community
  vni0001
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 from community
  vni0002
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 from community
  vni0005
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 from community
  vni0006
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 then accept
set policy-options community vni0001 members target:10003:1
set policy-options community vni0002 members target:10003:2
set policy-options community vni0005 members target:10003:5
set policy-options community vni0006 members target:10003:6
set routing-instances cust0001 instance-type vrf
set routing-instances cust0001 interface irb.1
set routing-instances cust0001 interface irb.2
set routing-instances cust0001 interface lo0.1
set routing-instances cust0001 route-distinguisher 10.0.0.10:2001
set routing-instances cust0001 vrf-target target:10001:1
set routing-instances cust0001 routing-options auto-export
set routing-instances cust0002 instance-type vrf
set routing-instances cust0002 interface irb.5
set routing-instances cust0002 interface irb.6
set routing-instances cust0002 interface lo0.2
set routing-instances cust0002 route-distinguisher 10.0.0.10:2002
```

```
set routing-instances cust0002 vrf-target target:10001:2
set routing-instances cust0002 routing-options auto-export
set vlans v0001 vlan-id 1
set vlans v0001 l3-interface irb.1
set vlans v0001 vxlan vni 1
set vlans v0001 vxlan ingress-node-replication
set vlans v0002 vlan-id 2
set vlans v0002 l3-interface irb.2
set vlans v0002 vxlan vni 2
set vlans v0002 vxlan ingress-node-replication
set vlans v0005 vlan-id 5
set vlans v0005 l3-interface irb.5
set vlans v0005 vxlan vni 5
set vlans v0005 vxlan ingress-node-replication
set vlans v0006 vlan-id 6
set vlans v0006 l3-interface irb.6
set vlans v0006 vxlan vni 6
set vlans v0006 vxlan ingress-node-replication
set policy-options policy-statement cust0001_vrf_imp term 1 from community cust0002
set policy-options policy-statement cust0001_vrf_imp term 1 then accept
set policy-options policy-statement cust0002_vrf_imp term 1 from community cust0001
set policy-options policy-statement cust0002_vrf_imp term 1 then accept
set policy-options community cust0001 members target:10001:1
set policy-options community cust0002 members target:10001:2
set routing-instances cust0001 vrf-import cust0001_vrf_imp
set routing-instances cust0002 vrf-import cust0002_vrf_imp
```

Step-by-Step Procedure

To configure profiles for the two customer groups:

1. Configure the IRB interfaces that enable communication among VXLANs 1, 2, 5, and 6.

```
[edit interfaces]
user@switch# set irb unit 1 family inet address 10.2.1.10/24 virtual-gateway-address
  10.2.1.254
user@switch# set irb unit 2 family inet address 10.2.2.10/24 virtual-gateway-address
  10.2.2.254
user@switch# set irb unit 5 family inet address 10.2.5.10/24 virtual-gateway-address
  10.2.5.254
user@switch# set irb unit 6 family inet address 10.2.6.10/24 virtual-gateway-address
  10.2.6.254
```

2. Configure a loopback interface for each customer group.

```
[edit interfaces]
user@switch# set lo0 unit 1 family inet address 127.10.0.1/32
user@switch# set lo0 unit 2 family inet address 127.10.0.2/32
```

3. Specify which VNIs are included in the EVPN-VXLAN domains.

```
[edit protocols]
user@switch# set evpn extended-vni-list [ 1 2 5 6 ]
```

4. Configure a route target for each VNI.

```
[edit protocols]
user@switch# set evpn vni-options vni 1 vrf-target export target:10003:1
user@switch# set evpn vni-options vni 2 vrf-target export target:10003:2
user@switch# set evpn vni-options vni 5 vrf-target export target:10003:5
user@switch# set evpn vni-options vni 6 vrf-target export target:10003:6
```

5. Configure policies that accept and import overlay routes.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 from
  community cust0001
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 from
  community cust0002
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 from
  community vni0001
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 from
  community vni0002
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 from
  community vni0005
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 from
  community vni0006
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 then accept
```

6. Set up communities for the customer groups and VXLANs. .

```
[edit policy-options]
user@switch# set community vni0001 members target:10003:1
user@switch# set community vni0002 members target:10003:2
user@switch# set community vni0005 members target:10003:5
user@switch# set community vni0006 members target:10003:6
```

7. Set up a routing instance for customer group 1.

NOTE: When setting up a routing instance for EVPN-VXLAN, you must include a loopback interface. You must also specify an IP address for the interface using the set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix*. If you inadvertently omit the loopback interface and associated IP address, EVPN control packets cannot be processed.

```
[edit routing-instances]
user@switch# set cust0001 instance-type vrf
user@switch# set cust0001 interface irb.1
user@switch# set cust0001 interface irb.2
user@switch# set cust0001 interface lo0.1
user@switch# set cust0001 route-distinguisher 10.0.0.10:2001
user@switch# set cust0001 vrf-target target:10001:1
user@switch# set cust0001 routing-options auto-export
```

8. Set up a routing instance for customer group 2.

NOTE: When setting up a routing instance for EVPN-VXLAN, you must include a loopback interface. You must also specify an IP address for the interface using the set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix*. If you inadvertently omit the loopback interface and associated IP address, EVPN control packets cannot be processed.

```
[edit routing-instances]
user@switch# set cust0002 instance-type vrf
user@switch# set cust0002 interface irb.5
user@switch# set cust0002 interface irb.6
user@switch# set cust0002 interface lo0.2
user@switch# set cust0002 route-distinguisher 10.0.0.10:2002
user@switch# set cust0002 vrf-target target:10001:2
user@switch# set cust0002 routing-options auto-export
```

9. Configure VXLANs v0001, v0002, v0005, and v0006, and associate the corresponding VNIs and IRB interfaces with each VXLAN.

```
[edit vlans]
user@switch# set v0001 vlan-id 1
user@switch# set v0001 l3-interface irb.1
user@switch# set v0001 vxlan vni 1
user@switch# set v0001 vxlan ingress-node-replication
user@switch# set v0002 vlan-id 2
user@switch# set v0002 l3-interface irb.2
```

```
user@switch# set v0002 vxlan vni 2
user@switch# set v0002 vxlan ingress-node-replication
user@switch# set v0005 vlan-id 5
user@switch# set v0005 l3-interface irb.5
user@switch# set v0005 vxlan vni 5
user@switch# set v0005 vxlan ingress-node-replication
user@switch# set v0006 vlan-id 6
user@switch# set v0006 l3-interface irb.6
user@switch# set v0006 vxlan vni 6
user@switch# set v0006 vxlan ingress-node-replication
```

10. Configure the importing of routes from one customer VRF instance into the other.

```
[edit policy-options]
user@switch# set policy-statement cust0001_vrf_imp term 1 from community
  cust0002
user@switch# set policy-statement cust0001_vrf_imp term 1 then accept
user@switch# set policy-statement cust0002_vrf_imp term 1 from community
  cust0001
user@switch# set policy-statement cust0002_vrf_imp term 1 then accept
user@switch# set community cust0001 members target:10001:1
user@switch# set community cust0002 members target:10001:2
[edit routing-instances]
user@switch# set cust0001 vrf-import cust0001_vrf_imp
user@switch# set cust0002 vrf-import cust0002_vrf_imp
```

Spine 2: Underlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces et-0/0/0 vlan-tagging
set interfaces et-0/0/0 unit 0 vlan-id 1
set interfaces et-0/0/0 unit 0 family inet address 10.1.12.9/24
set interfaces et-0/0/2 unit 0 family inet address 10.1.13.9/24
set interfaces et-0/0/8 unit 0 family inet address 10.1.4.9/24
set interfaces et-0/0/9 unit 0 family inet address 10.1.40.9/24
set interfaces et-0/0/10 unit 0 family inet address 10.1.5.9/24
set interfaces et-0/0/11 unit 0 family inet address 10.1.41.9/24
set interfaces lo0 unit 0 family inet address 10.0.0.9/32 primary
set routing-options router-id 10.0.0.9
set routing-options autonomous-system 65001
set routing-options forwarding-table export load-balancing-policy
set protocols bgp multihop
set protocols bgp group underlay type external
set protocols bgp group underlay accept-remote-nexthop
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 5
```

```
set protocols bgp group underlay export send-direct
set protocols bgp group underlay peer-as 65000
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.1.4.7
set protocols bgp group underlay neighbor 10.1.5.8
set protocols bgp group underlay neighbor 10.1.12.3
set protocols bgp group underlay neighbor 10.1.13.4
set protocols bgp group underlay neighbor 10.1.40.7
set protocols bgp group underlay neighbor 10.1.41.8
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 10 from protocol direct
set policy-options policy-statement send-direct term 10 then accept
```

Configuring the Underlay Network on Spine 2

Step-by-Step Procedure

To configure the underlay network on Spine 2:

1. Configure Layer 2 and Layer 3 interfaces.

   ```
   [edit interfaces]
   user@switch#set et-0/0/0 vlan-tagging
   user@switch#set et-0/0/0 unit 0 vlan-id 1
   user@switch#set et-0/0/0 unit 0 family inet address 10.1.12.9/24
   user@switch#set et-0/0/2 unit 0 family inet address 10.1.13.9/24
   user@switch#set et-0/0/8 unit 0 family inet address 10.1.4.9/24
   user@switch#set et-0/0/9 unit 0 family inet address 10.1.40.9/24
   user@switch#set et-0/0/10 unit 0 family inet address 10.1.5.9/24
   user@switch#set et-0/0/11 unit 0 family inet address 10.1.41.9/24
   ```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

   ```
   [edit interfaces]
   user@switch# set lo0 unit 0 family inet address 10.0.0.9/32 primary
   ```

3. Set the routing options.

   ```
   [edit routing-options]
   user@switch# set router-id 10.0.0.9
   user@switch# set autonomous-system 65001
   user@switch# set forwarding-table export load-balancing-policy
   ```

4. Configure multihop EBGP.

   ```
   [edit protocols]
   user@switch# set bgp multihop
   ```

5. Configure a BGP group that includes peers that handle underlay functions.

```
[edit protocols]
user@switch# set bgp group underlay type external
user@switch# set bgp group underlay accept-remote-nexthop
user@switch# set bgp group underlay advertise-peer-as
user@switch# set bgp group underlay family inet unicast loops 5
user@switch# set bgp group underlay export send-direct
user@switch# set bgp group underlay peer-as 65000
user@switch# set bgp group underlay multipath
user@switch# set bgp group underlay neighbor 10.1.4.7
user@switch# set bgp group underlay neighbor 10.1.5.8
user@switch# set bgp group underlay neighbor 10.1.12.3
user@switch# set bgp group underlay neighbor 10.1.13.4
user@switch# set bgp group underlay neighbor 10.1.40.7
user@switch# set bgp group underlay neighbor 10.1.41.8
```

NOTE: You must enter the **set protocols bgp group underlay accept-remote-nexthop** command so that the BGP routes appear in the routing table as direct routes. VXLAN requires direct next hops for equal-cost multipath (ECMP) load balancing.

6. Configure a policy that spreads traffic across multiple paths between the Juniper Networks switches.

```
[edit policy-options]
user@switch# set policy-statement load-balancing-policy then load-balance
   per-packet
```

7. Configure a policy that enables BGP to advertise direct interfaces such as loopback interface lo0.

```
[edit policy-options]
user@switch# set policy-statement send-direct term 10 from protocol direct
user@switch# set policy-statement send-direct term 10 then accept
```

Spine 2: Overlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.0.0.9
```

```
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.0.0.7
set protocols bgp group evpn neighbor 10.0.0.8
set protocols bgp group evpn neighbor 10.0.0.3
set protocols bgp group evpn neighbor 10.0.0.4
set protocols bgp group evpn neighbor 10.0.0.10
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
set policy-options community switch_options_comm members target:65000:2
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.0.0.9:1
set switch-options vrf-import EVPN_VRF_IMPORT
set switch-options vrf-target target:65000:2
set switch-options vrf-target auto
```

Configuring the Overlay Network on Spine 2

Step-by-Step Procedure

To configure the overlay network on Spine 2:

1. Configure an IBGP group for the EVPN-VXLAN overlay.

```
[edit protocols]
user@switch# set bgp group evpn type internal
user@switch# set bgp group evpn local-address 10.0.0.9
user@switch# set bgp group evpn family evpn signaling
user@switch# set bgp group evpn local-as 65000
user@switch# set bgp group evpn multipath
user@switch# set bgp group evpn neighbor 10.0.0.7
user@switch# set bgp group evpn neighbor 10.0.0.8
user@switch# set bgp group evpn neighbor 10.0.0.10
user@switch# set bgp group evpn neighbor 10.0.0.4
user@switch# set bgp group evpn neighbor 10.0.0.3
```

2. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors.

```
[edit protocols]
user@switch# set evpn encapsulation vxlan
```

3. Specify a tunnel type used for passing multicast traffic between Juniper Networks switches in an EVPN-VXLAN environment.

```
[edit protocols]
user@switch# set evpn multicast-mode ingress-replication
```

4. Set up a policy to handle Layer 2 EVPN traffic.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
user@switch# set community switch_options_comm members target:65000:2
```

5. Configure options for the default routing instance (virtual switch type).

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 10.0.0.9:1
user@switch# set vrf-import EVPN_VRF_IMPORT
user@switch# set vrf-target target:65000:2
user@switch# set vrf-target auto
```

Spine 2: Customer Profile Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

NOTE: When setting up a routing instance for EVPN-VXLAN, you must include a loopback interface. You must also specify an IP address for the interface using the set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix*. If you inadvertently omit the loopback interface and associated IP address, EVPN control packets cannot be processed.

```
set interfaces irb unit 1 family inet address 10.2.1.9/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.9/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 5 family inet address 10.2.5.9/24 virtual-gateway-address 10.2.5.254
set interfaces irb unit 6 family inet address 10.2.6.9/24 virtual-gateway-address 10.2.6.254
set interfaces lo0 unit 1 family inet address 127.9.0.1/32
set interfaces lo0 unit 2 family inet address 127.9.0.2/32
set protocols evpn extended-vni-list [ 1 2 5 6 ]
set protocols evpn vni-options vni 1 vrf-target export target:10003:1
set protocols evpn vni-options vni 2 vrf-target export target:10003:2
set protocols evpn vni-options vni 5 vrf-target export target:10003:5
set protocols evpn vni-options vni 6 vrf-target export target:10003:6
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 from community
  cust0001
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 from community
  cust0002
```

```
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 from community
  vni0001
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 from community
  vni0002
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 from community
  vni0005
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 from community
  vni0006
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 then accept
set policy-options community vni0001 members target:10003:1
set policy-options community vni0002 members target:10003:2
set policy-options community vni0005 members target:10003:5
set policy-options community vni0006 members target:10003:6
set routing-instances cust0001 instance-type vrf
set routing-instances cust0001 interface irb.1
set routing-instances cust0001 interface irb.2
set routing-instances cust0001 interface lo0.1
set routing-instances cust0001 route-distinguisher 10.0.0.9:2001
set routing-instances cust0001 vrf-target target:10001:1
set routing-instances cust0001 routing-options auto-export
set routing-instances cust0002 instance-type vrf
set routing-instances cust0002 interface irb.5
set routing-instances cust0002 interface irb.6
set routing-instances cust0002 interface lo0.2
set routing-instances cust0002 route-distinguisher 10.0.0.9:2002
set routing-instances cust0002 vrf-target target:10001:2
set routing-instances cust0002 routing-options auto-export
set vlans v0001 vlan-id 1
set vlans v0001 l3-interface irb.1
set vlans v0001 vxlan vni 1
set vlans v0001 vxlan ingress-node-replication
set vlans v0002 vlan-id 2
set vlans v0002 l3-interface irb.2
set vlans v0002 vxlan vni 2
set vlans v0002 vxlan ingress-node-replication
set vlans v0005 vlan-id 5
set vlans v0005 l3-interface irb.5
set vlans v0005 vxlan vni 5
set vlans v0005 vxlan ingress-node-replication
set vlans v0006 vlan-id 6
set vlans v0006 l3-interface irb.6
set vlans v0006 vxlan vni 6
set vlans v0006 vxlan ingress-node-replication
set policy-options policy-statement cust0001_vrf_imp term 1 from community cust0002
set policy-options policy-statement cust0001_vrf_imp term 1 then accept
set policy-options policy-statement cust0002_vrf_imp term 1 from community cust0001
set policy-options policy-statement cust0002_vrf_imp term 1 then accept
set policy-options community cust0001 members target:10001:1
set policy-options community cust0002 members target:10001:2
set routing-instances cust0001 vrf-import cust0001_vrf_imp
set routing-instances cust0002 vrf-import cust0002_vrf_imp
```

Step-by-Step Procedure

To configure profiles for the two customer groups:

1. Configure the IRB interfaces that enable communication among VXLANs v0001, v0002, v0005, and v0006.

```
[edit interfaces]
user@switch# set irb unit 1 family inet address 10.2.1.9/24 virtual-gateway-address
  10.2.1.254
user@switch# set irb unit 2 family inet address 10.2.2.9/24 virtual-gateway-address
  10.2.2.254
user@switch# set irb unit 5 family inet address 10.2.5.9/24 virtual-gateway-address
  10.2.5.254
user@switch# set irb unit 6 family inet address 10.2.6.9/24 virtual-gateway-address
  10.2.6.254
```

2. Configure a loopback interface for each customer group.

```
[edit interfaces]
user@switch# set lo0 unit 1 family inet address 127.9.0.1/32
user@switch# set lo0 unit 2 family inet address 127.9.0.2/32
```

3. Specify which VNIs are included in the EVPN-VXLAN domains.

```
[edit protocols]
user@switch# set evpn extended-vni-list [ 1 2 5 6 ]
```

4. Configure a route target for each VNI.

```
[edit protocols]
user@switch# set evpn vni-options vni 1 vrf-target export target:10003:1
user@switch# set evpn vni-options vni 2 vrf-target export target:10003:2
user@switch# set evpn vni-options vni 5 vrf-target export target:10003:5
user@switch# set evpn vni-options vni 6 vrf-target export target:10003:6
```

5. Configure policies that accept and import overlay routes.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 from
  community cust0001
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 from
  community cust0002
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 from
  community vni0001
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 from
  community vni0002
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 then accept
```

```
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 from
  community vni0005
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 from
  community vni0006
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 then accept
```

6. Set up communities for the customer groups and VXLANs.

```
user@switch# set community vni0001 members target:10003:1
user@switch# set community vni0002 members target:10003:2
user@switch# set community vni0005 members target:10003:5
user@switch# set community vni0006 members target:10003:6
```

7. Set up a routing instance for customer 1.

NOTE: When setting up a routing instance for EVPN-VXLAN, you must include a loopback interface. You must also specify an IP address for the interface using the set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix*. If you inadvertently omit the loopback interface and associated IP address, EVPN control packets cannot be processed.

```
[edit routing-instances]
user@switch# set cust0001 instance-type vrf
user@switch# set cust0001 interface irb.1
user@switch# set cust0001 interface irb.2
user@switch# set cust0001 interface lo0.1
user@switch# set cust0001 route-distinguisher 10.0.0.9:2001
user@switch# set cust0001 vrf-target target:10001:1
user@switch# set cust0001 routing-options auto-export
```

8. Set up a routing instance for customer 2.

NOTE: When setting up a routing instance for EVPN-VXLAN, you must include a loopback interface. You must also specify an IP address for the interface using the set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix*. If you inadvertently omit the loopback interface and associated IP address, EVPN control packets cannot be processed.

```
[edit routing-instances]
user@switch# set cust0002 instance-type vrf
user@switch# set cust0002 interface irb.5
```

```
user@switch# set cust0002 interface irb.6
user@switch# set cust0002 interface lo0.2
user@switch# set cust0002 route-distinguisher 10.0.0.9:2002
user@switch# set cust0002 vrf-target target:10001:2
user@switch# set cust0002 routing-options auto-export
```

9. Configure VXLANs v0001, v0002, v0005, and v0006, and associate the corresponding VNIs and IRB interfaces with each VXLAN.

```
[edit vlans]
user@switch# set v0001 vlan-id 1
user@switch# set v0001 l3-interface irb.1
user@switch# set v0001 vxlan vni 1
user@switch# set v0001 vxlan ingress-node-replication
user@switch# set v0002 vlan-id 2
user@switch# set v0002 l3-interface irb.2
user@switch# set v0002 vxlan vni 2
user@switch# set v0002 vxlan ingress-node-replication
user@switch# set v0005 vlan-id 5
user@switch# set v0005 l3-interface irb.5
user@switch# set v0005 vxlan vni 5
user@switch# set v0005 vxlan ingress-node-replication
user@switch# set v0006 vlan-id 6
user@switch# set v0006 l3-interface irb.6
user@switch# set v0006 vxlan vni 6
user@switch# set v0006 vxlan ingress-node-replication
```

10. Configure the importing of routes from one customer VRF instance into the other.

```
[edit policy-options]
user@switch# set policy-statement cust0001_vrf_imp term 1 from community
  cust0002
user@switch# set policy-statement cust0001_vrf_imp term 1 then accept
user@switch# set policy-statement cust0002_vrf_imp term 1 from community
  cust0001
user@switch# set policy-statement cust0002_vrf_imp term 1 then accept
user@switch# set community cust0001 members target:10001:1
user@switch# set community cust0002 members target:10001:2
[edit routing-instances]
user@switch# set cust0001 vrf-import cust0001_vrf_imp
user@switch# set cust0002 vrf-import cust0002_vrf_imp
```

Leaf 1: Underlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces et-0/0/49 vlan-tagging
set interfaces et-0/0/49 unit 0 vlan-id 1
set interfaces et-0/0/49 unit 0 family inet address 10.1.12.3/24
set interfaces et-0/0/50 vlan-tagging
set interfaces et-0/0/50 unit 0 vlan-id 1
set interfaces et-0/0/50 unit 0 family inet address 10.1.24.3/24
set interfaces et-0/0/51 vlan-tagging
set interfaces et-0/0/51 unit 0 vlan-id 1
set interfaces et-0/0/51 unit 0 family inet address 10.1.30.3/24
set interfaces lo0 unit 0 family inet address 10.0.0.3/32 primary
set routing-options router-id 10.0.0.3
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp multihop
set protocols bgp group underlay type external
set protocols bgp group underlay accept-remote-nexthop
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 5
set protocols bgp group underlay export send-direct
set protocols bgp group underlay peer-as 65001
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.1.12.9
set protocols bgp group underlay neighbor 10.1.24.10
set protocols bgp group underlay neighbor 10.1.30.10
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 10 from protocol direct
set policy-options policy-statement send-direct term 10 then accept
```

Configuring the Underlay Network for Leaf 1

Step-by-Step Procedure

To configure the underlay network for Leaf 1:

1. Configure Layer 2 interfaces and associate each interface with VXLAN v0001.

```
[edit interfaces]
user@switch# set et-0/0/49 vlan-tagging
user@switch# set et-0/0/49 unit 0 vlan-id 1
user@switch# set et-0/0/49 unit 0 family inet address 10.1.12.3/24
user@switch# set et-0/0/50 vlan-tagging
user@switch# set et-0/0/50 unit 0 vlan-id 1
user@switch# set et-0/0/50 unit 0 family inet address 10.1.24.3/24
user@switch# set et-0/0/51 vlan-tagging
user@switch# set et-0/0/51 unit 0 vlan-id 1
user@switch# set et-0/0/51 unit 0 family inet address 10.1.30.3/24
```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.0.0.3/32 primary
```

3. Set the routing options.

```
[edit routing-options]
user@switch# set router-id 10.0.0.3
user@switch# set autonomous-system 65000
user@switch# set forwarding-table export load-balancing-policy
```

4. Configure multihop EBGP.

```
[edit protocols]
user@switch# set bgp multihop
```

5. Configure a BGP group that includes peers that handle underlay functions.

```
[edit protocols]
user@switch# set bgp group underlay type external
user@switch# set bgp group underlay accept-remote-nexthop
user@switch# set bgp group underlay advertise-peer-as
user@switch# set bgp group underlay family inet unicast loops 5
user@switch# set bgp group underlay export send-direct
user@switch# set bgp group underlay peer-as 65001
user@switch# set bgp group underlay multipath
user@switch# set bgp group underlay neighbor 10.1.12.9
user@switch# set bgp group underlay neighbor 10.1.24.10
user@switch# set bgp group underlay neighbor 10.1.30.10
```

NOTE: You must enter the **set protocols bgp group underlay accept-remote-nexthop** command so that the BGP routes appear in the routing table as direct routes. VXLAN requires direct next hops for equal-cost multipath (ECMP) load balancing.

6. Configure a policy that spreads traffic across multiple paths between the Juniper Networks switches.

```
[edit policy-options]
user@switch# set policy-statement load-balancing-policy then load-balance
  per-packet
```

7. Configure a policy that enables BGP to advertise direct interfaces such as loopback interface lo0.

```
[edit policy-options]
user@switch# set policy-statement send-direct term 10 from protocol direct
user@switch# set policy-statement send-direct term 10 then accept
```

Leaf 1: Overlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.0.0.3
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.0.0.4
set protocols bgp group evpn neighbor 10.0.0.7
set protocols bgp group evpn neighbor 10.0.0.8
set protocols bgp group evpn neighbor 10.0.0.9
set protocols bgp group evpn neighbor 10.0.0.10
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
set policy-options community switch_options_comm members target:65000:2
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.0.0.3:1
set switch-options vrf-import EVPN_VRF_IMPORT
set switch-options vrf-target target:65000:2
set switch-options vrf-target auto
```

Configuring the Overlay Network for Leaf 1

Step-by-Step Procedure

To configure the overlay network for Leaf 1:

1. Configure an IBGP group for the EVPN-VXLAN overlay network.

```
[edit protocols]
user@switch# set bgp group evpn type internal
user@switch# set bgp group evpn local-address 10.0.0.3
user@switch# set bgp group evpn family evpn signaling
user@switch# set bgp group evpn multipath
user@switch# set bgp group evpn neighbor 10.0.0.4
user@switch# set bgp group evpn neighbor 10.0.0.7
user@switch# set bgp group evpn neighbor 10.0.0.8
user@switch# set bgp group evpn neighbor 10.0.0.9
user@switch# set bgp group evpn neighbor 10.0.0.10
```

2. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors.

```
[edit protocols]
```

```
user@switch# set evpn encapsulation vxlan
```

3. Specify a tunnel type used for passing multicast traffic between Juniper Networks switches in an EVPN-VXLAN environment.

```
[edit protocols]
user@switch# set evpn multicast-mode ingress-replication
```

4. Set up a policy to handle Layer 2 EVPN traffic.

```
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
user@switch# set community switch_options_comm members target:65000:2
```

5. Configure options for the default routing instance (virtual switch type).

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 10.0.0.3:1
user@switch# set vrf-import EVPN_VRF_IMPORT
user@switch# set vrf-target target:65000:2
user@switch# set vrf-target auto
```

Leaf 1: Customer Profile Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:00:00:ab:cd:00:01:00:00:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:11:00:00:00:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
set protocols evpn vni-options vni 1 vrf-target export target:10003:1
set protocols evpn vni-options vni 2 vrf-target export target:10003:2
set protocols evpn vni-options vni 5 vrf-target export target:10003:5
set protocols evpn vni-options vni 6 vrf-target export target:10003:6
set protocols evpn extended-vni-list [ 1 2 5 6 ]
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 from community
  cust0001
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 then accept
```

```
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 from community
  cust0002
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 from community
  vni0001
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 from community
  vni0002
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 from community
  vni0005
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 from community
  vni0006
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 then accept
set policy-options community cust0001 members target:10001:1
set policy-options community cust0002 members target:10001:2
set policy-options community vni0001 members target:10003:1
set policy-options community vni0002 members target:10003:2
set policy-options community vni0005 members target:10003:5
set policy-options community vni0006 members target:10003:6
set vlans v0001 vlan-id 1
set vlans v0001 vxlan vni 1
set vlans v0001 vxlan ingress-node-replication
set vlans v0002 vlan-id 2
set vlans v0002 vxlan vni 2
set vlans v0002 vxlan ingress-node-replication
set vlans v0005 vlan-id 5
set vlans v0005 vxlan vni 5
set vlans v0005 vxlan ingress-node-replication
set vlans v0006 vlan-id 6
set vlans v0006 vxlan vni 6
set vlans v0006 vxlan ingress-node-replication
```

Configuring the Customer Profiles for Leaf 1

Step-by-Step Procedure

To configure the customer profiles for the two customer groups:

1. Configure an aggregated Ethernet interface for the connection with the physical server. This interface is associated with VXLANs v0001, v0002, v0005, and v0006.

```
[edit interfaces]
user@switch# set xe-0/0/0 ether-options 802.3ad ae0
user@switch# set ae0 esi 00:00:00:ab:cd:00:01:00:00:01
user@switch# set ae0 esi all-active
user@switch# set ae0 aggregated-ether-options lacp active
user@switch# set ae0 aggregated-ether-options lacp system-id 00:11:00:00:00:01
user@switch# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@switch# set ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
```

2. Configure a route target for each VNI.

```
[edit protocols]
user@switch# set evpn vni-options vni 1 vrf-target export target:10003:1
user@switch# set evpn vni-options vni 2 vrf-target export target:10003:2
user@switch# set evpn vni-options vni 5 vrf-target export target:10003:5
user@switch# set evpn vni-options vni 6 vrf-target export target:10003:6
```

3. Specify which VNIs are included in the EVPN-VXLAN domains.

```
[edit protocols]
user@switch# set evpn extended-vni-list [ 1 2 5 6 ]
```

4. Configure policies that accept and import overlay routes.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 from
  community cust0001
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 from
  community cust0002
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 from
  community vni0001
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 from
  community vni0002
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 from
  community vni0005
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 from
  community vni0006
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 then accept
```

5. Set up communities for the customer groups and VXLANs.

```
[edit policy-options]
user@switch# set community cust0001 members target:10001:1
user@switch# set community cust0002 members target:10001:2
user@switch# set community vni0001 members target:10003:1
user@switch# set community vni0002 members target:10003:2
user@switch# set community vni0005 members target:10003:5
user@switch# set community vni0006 members target:10003:6
```

6. Configure VXLANs v0001, v0002, v0005, and v0006.

```
[edit vlans]
user@switch# set v0001 vlan-id 1
```

```
user@switch# set v0001 vxlan vni 1
user@switch# set v0001 vxlan ingress-node-replication
user@switch# set v0002 vlan-id 2
user@switch# set v0002 vxlan vni 2
user@switch# set v0002 vxlan ingress-node-replication
user@switch# set v0005 vlan-id 5
user@switch# set v0005 vxlan vni 5
user@switch# set v0005 vxlan ingress-node-replication
user@switch# set v0006 vlan-id 6
user@switch# set v0006 vxlan vni 6
user@switch# set v0006 vxlan ingress-node-replication
```

Leaf 2: Underlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces et-0/0/49 unit 0 family inet address 10.1.13.4/24
set interfaces et-0/0/50 unit 0 family inet address 10.1.25.4/24
set interfaces et-0/0/51 unit 0 family inet address 10.1.31.4/24
set interfaces lo0 unit 0 family inet address 10.0.0.4/32 primary
set routing-options router-id 10.0.0.4
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp multihop
set protocols bgp group underlay type external
set protocols bgp group underlay accept-remote-nexthop
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 5
set protocols bgp group underlay export send-direct
set protocols bgp group underlay peer-as 65001
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.1.13.9
set protocols bgp group underlay neighbor 10.1.25.10
set protocols bgp group underlay neighbor 10.1.31.10
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 10 from protocol direct
set policy-options policy-statement send-direct term 10 then accept
```

Configuring the Underlay Network for Leaf 2

Step-by-Step Procedure

To configure the underlay network for Leaf 2:

1. Configure Layer 2 and Layer 3 interfaces.

```
[edit interfaces]
set et-0/0/49 unit 0 family inet address 10.1.13.4/24
set et-0/0/50 unit 0 family inet address 10.1.25.4/24
```

```
set et-0/0/51 unit 0 family inet address 10.1.31.4/24
```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.0.0.4/32 primary
```

3. Set the routing options.

```
[edit routing-options]
user@switch# set router-id 10.0.0.4
user@switch# set autonomous-system 65000
user@switch# set forwarding-table export load-balancing-policy
```

4. Configure multihop EBGP.

```
[edit protocols]
user@switch# set bgp multihop
```

5. Configure a BGP group that includes peers that handle underlay functions.

```
[edit protocols]
user@switch# set bgp group underlay type external
user@switch# set bgp group underlay accept-remote-nexthop
user@switch# set bgp group underlay advertise-peer-as
user@switch# set bgp group underlay family inet unicast loops 5
user@switch# set bgp group underlay export send-direct
user@switch# set bgp group underlay peer-as 65001
user@switch# set bgp group underlay multipath
user@switch# set bgp group underlay neighbor 10.1.13.9
user@switch# set bgp group underlay neighbor 10.1.25.10
user@switch# set bgp group underlay neighbor 10.1.31.10
```

NOTE: You must enter the **set protocols bgp group underlay accept-remote-nexthop** command so that the BGP routes appear in the routing table as direct routes. VXLAN requires direct next hops for equal-cost multipath (ECMP) load balancing.

6. Configure a policy that spreads traffic across multiple paths between the Juniper Networks switches.

```
[edit policy-options]
```

```
user@switch# set policy-statement load-balancing-policy then load-balance
  per-packet
```

7. Configure a policy that enables BGP to advertise direct interfaces such as loopback interface lo0.

```
[edit policy-options]
user@switch# set policy-statement send-direct term 10 from protocol direct
user@switch# set policy-statement send-direct term 10 then accept
```

Leaf 2: Overlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.0.0.4
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.0.0.3
set protocols bgp group evpn neighbor 10.0.0.7
set protocols bgp group evpn neighbor 10.0.0.8
set protocols bgp group evpn neighbor 10.0.0.9
set protocols bgp group evpn neighbor 10.0.0.10
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
set policy-options community switch_options_comm members target:65000:2
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.0.0.4:1
set switch-options vrf-import EVPN_VRF_IMPORT
set switch-options vrf-target target:65000:2
set switch-options vrf-target auto
```

Configuring the Overlay Network for Leaf 2

Step-by-Step Procedure

To configure the overlay network for Leaf 2:

1. Configure an IBGP group for the EVPN-VXLAN overlay network.

```
[edit protocols]
user@switch# set bgp group evpn type internal
user@switch# set bgp group evpn local-address 10.0.0.4
user@switch# set bgp group evpn family evpn signaling
```

```
user@switch# set bgp group evpn multipath
user@switch# set bgp group evpn neighbor 10.0.0.3
user@switch# set bgp group evpn neighbor 10.0.0.7
user@switch# set bgp group evpn neighbor 10.0.0.8
user@switch# set bgp group evpn neighbor 10.0.0.9
user@switch# set bgp group evpn neighbor 10.0.0.10
```

2. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors.

```
[edit protocols]
user@switch# set evpn encapsulation vxlan
```

3. Specify a tunnel type used for passing multicast traffic between Juniper Networks switches in an EVPN-VXLAN environment.

```
[edit protocols]
user@switch# set evpn multicast-mode ingress-replication
```

4. Set up a policy to handle Layer 2 EVPN traffic.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
   from community switch_options_comm
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
   then accept
user@switch# set community switch_options_comm members target:65000:2
```

5. Configure options for the default routing instance (virtual switch type).

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 10.0.0.4:1
user@switch# set vrf-import EVPN_VRF_IMPORT
user@switch# set vrf-target target:65000:2
user@switch# set vrf-target auto
```

Leaf 2: Customer Profile Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:00:00:ab:cd:00:01:00:00:01
```

```
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:11:00:00:00:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
set protocols evpn vni-options vni 1 vrf-target export target:10003:1
set protocols evpn vni-options vni 2 vrf-target export target:10003:2
set protocols evpn vni-options vni 5 vrf-target export target:10003:5
set protocols evpn vni-options vni 6 vrf-target export target:10003:6
set protocols evpn extended-vni-list [ 1 2 5 6 ]
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 from community
  cust0001
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 from community
  cust0002
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 from community
  vni0001
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 from community
  vni0002
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 from community
  vni0005
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 from community
  vni0006
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 then accept
set policy-options community cust0001 members target:10001:1
set policy-options community cust0002 members target:10001:2
set policy-options community vni0001 members target:10003:1
set policy-options community vni0002 members target:10003:2
set policy-options community vni0005 members target:10003:5
set policy-options community vni0006 members target:10003:6
set vlans v0001 vlan-id 1
set vlans v0001 vxlan vni 1
set vlans v0001 vxlan ingress-node-replication
set vlans v0002 vlan-id 2
set vlans v0002 vxlan vni 2
set vlans v0002 vxlan ingress-node-replication
set vlans v0005 vlan-id 5
set vlans v0005 vxlan vni 5
set vlans v0005 vxlan ingress-node-replication
set vlans v0006 vlan-id 6
set vlans v0006 vxlan vni 6
set vlans v0006 vxlan ingress-node-replication
```

Configuring the Customer Profiles for Leaf 2

Step-by-Step Procedure

To configure profiles for the two customer groups:

1. Configure an aggregated Ethernet interface for the connection with the physical server. This interface is associated with VXLANs v0001, v0002, v0005, and v0006.

```
[edit interfaces]
user@switch# set xe-0/0/0 ether-options 802.3ad ae0
user@switch# set ae0 esi 00:00:00:ab:cd:00:01:00:00:01
user@switch# set ae0 esi all-active
user@switch# set ae0 aggregated-ether-options lacp active
user@switch# set ae0 aggregated-ether-options lacp system-id 00:11:00:00:00:01
user@switch# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@switch# set ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
```

2. Configure a route target for each VNI.

```
[edit protocols]
user@switch# set evpn vni-options vni 1 vrf-target export target:10003:1
user@switch# set evpn vni-options vni 2 vrf-target export target:10003:2
user@switch# set evpn vni-options vni 5 vrf-target export target:10003:5
user@switch# set evpn vni-options vni 6 vrf-target export target:10003:6
```

3. Specify which VNIs are included in the EVPN-VXLAN domains.

```
[edit protocols]
user@switch# set evpn extended-vni-list [ 1 2 5 6 ]
```

4. Configure policies that accept and import overlay routes.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 from
  community cust0001
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 from
  community cust0002
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 from
  community vni0001
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 from
  community vni0002
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 from
  community vni0005
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 from
  community vni0006
```

```
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 then accept
```

5. Set up the communities.

```
[edit policy-options]
user@switch# set community cust0001 members target:10001:1
user@switch# set community cust0002 members target:10001:2
user@switch# set community vni0001 members target:10003:1
user@switch# set community vni0002 members target:10003:2
user@switch# set community vni0005 members target:10003:5
user@switch# set community vni0006 members target:10003:6
```

6. Configure VXLANs v0001, v0002, v0005, and v0006.

```
[edit vlans]
user@switch# set v0001 vlan-id 1
user@switch# set v0001 vxlan vni 1
user@switch# set v0001 vxlan ingress-node-replication
user@switch# set v0002 vlan-id 2
user@switch# set v0002 vxlan vni 2
user@switch# set v0002 vxlan ingress-node-replication
user@switch# set v0005 vlan-id 5
user@switch# set v0005 vxlan vni 5
user@switch# set v0005 vxlan ingress-node-replication
user@switch# set v0006 vlan-id 6
user@switch# set v0006 vxlan vni 6
user@switch# set v0006 vxlan ingress-node-replication
```

Leaf 3: Underlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces et-0/0/48 unit 0 family inet address 10.1.4.7/24
set interfaces et-0/0/50 unit 0 family inet address 10.1.28.7/24
set interfaces et-0/0/51 unit 0 family inet address 10.1.34.7/24
set interfaces et-0/0/52 unit 0 family inet address 10.1.40.7/24
set interfaces lo0 unit 0 family inet address 10.0.0.7/32 primary
set routing-options router-id 10.0.0.7
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp multihop
set protocols bgp group underlay type external
set protocols bgp group underlay accept-remote-nexthop
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 5
set protocols bgp group underlay export send-direct
```

```
set protocols bgp group underlay peer-as 65001
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.1.4.9
set protocols bgp group underlay neighbor 10.1.28.10
set protocols bgp group underlay neighbor 10.1.34.10
set protocols bgp group underlay neighbor 10.1.40.9
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 10 from protocol direct
set policy-options policy-statement send-direct term 10 then accept
```

Configuring the Underlay Network for Leaf 3

Step-by-Step Procedure

To configure the underlay network for Leaf 3:

1. Configure Layer 3 interfaces.

   ```
   [edit interfaces]
   user@switch# set et-0/0/48 unit 0 family inet address 10.1.4.7/24
   user@switch# set et-0/0/50 unit 0 family inet address 10.1.28.7/24
   user@switch# set et-0/0/51 unit 0 family inet address 10.1.34.7/24
   user@switch# set et-0/0/52 unit 0 family inet address 10.1.40.7/24
   ```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

   ```
   [edit interfaces]
   user@switch# set lo0 unit 0 family inet address 10.0.0.7/32 primary
   ```

3. Set the routing options.

   ```
   [edit routing-options]
   user@switch# set router-id 10.0.0.7
   user@switch# set autonomous-system 65000
   user@switch# set forwarding-table export load-balancing-policy
   ```

4. Configure multihop EBGP.

   ```
   [edit protocols]
   user@switch# set bgp multihop
   ```

5. Configure a BGP group that includes peers that handle underlay functions.

   ```
   [edit protocols]
   user@switch# set bgp group underlay type external
   user@switch# set bgp group underlay accept-remote-nexthop
   user@switch# set bgp group underlay advertise-peer-as
   user@switch# set bgp group underlay family inet unicast loops 5
   ```

```
user@switch# set bgp group underlay export send-direct
user@switch# set bgp group underlay peer-as 65001
user@switch# set bgp group underlay multipath
user@switch# set bgp group underlay neighbor 10.1.4.9
user@switch# set bgp group underlay neighbor 10.1.28.10
user@switch# set bgp group underlay neighbor 10.1.34.10
user@switch# set bgp group underlay neighbor 10.1.40.9
```

NOTE: You must enter the **set protocols bgp group underlay accept-remote-nexthop** command so that the BGP routes appear in the routing table as direct routes. VXLAN requires direct next hops for equal-cost multipath (ECMP) load balancing.

6. Configure a policy that spreads traffic across multiple paths between the Juniper Networks switches.

```
[edit policy-options]
user@switch# set policy-statement load-balancing-policy then load-balance
  per-packet
```

7. Configure a policy that enables BGP to advertise direct interfaces such as loopback interface lo0.

```
[edit policy-options]
user@switch# set policy-statement send-direct term 10 from protocol direct
user@switch# set policy-statement send-direct term 10 then accept
```

Leaf 3: Overlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 10.0.0.7
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.0.0.3
set protocols bgp group evpn neighbor 10.0.0.4
set protocols bgp group evpn neighbor 10.0.0.8
set protocols bgp group evpn neighbor 10.0.0.9
set protocols bgp group evpn neighbor 10.0.0.10
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
```

```
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
   from community switch_options_comm
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
   then accept
set policy-options community switch_options_comm members target:65000:2
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.0.0.7:1
set switch-options vrf-import EVPN_VRF_IMPORT
set switch-options vrf-target target:65000:2
set switch-options vrf-target auto
```

Configuring the Overlay Network for Leaf 3

Step-by-Step Procedure

To configure the overlay network for Leaf 3:

1. Configure an IBGP group for the EVPN-VXLAN overlay network.

   ```
   [edit protocols]
   user@switch# set bgp group evpn type internal
   user@switch# set bgp group evpn local-address 10.0.0.7
   user@switch# set bgp group evpn family evpn signaling
   user@switch# set bgp group evpn multipath
   user@switch# set bgp group evpn neighbor 10.0.0.3
   user@switch# set bgp group evpn neighbor 10.0.0.4
   user@switch# set bgp group evpn neighbor 10.0.0.8
   user@switch# set bgp group evpn neighbor 10.0.0.9
   user@switch# set bgp group evpn neighbor 10.0.0.10
   ```

2. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors.

   ```
   [edit protocols]
   user@switch# set evpn encapsulation vxlan
   ```

3. Specify a tunnel type used for passing multicast traffic between Juniper Networks switches in an EVPN-VXLAN environment.

   ```
   [edit protocols]
   user@switch# set evpn multicast-mode ingress-replication
   ```

4. Set up a policy to handle Layer 2 EVPN traffic.

   ```
   [edit policy-options]
   user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
      from community switch_options_comm
   user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
      then accept
   user@switch# set community switch_options_comm members target:65000:2
   ```

5. Configure options for the default routing instance (virtual switch type).

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 10.0.0.7:1
user@switch# set vrf-import EVPN_VRF_IMPORT
user@switch# set vrf-target target:65000:2
user@switch# set vrf-target auto
```

Leaf 3: Customer Profile Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:00:00:ab:cd:00:02:00:00:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:22:00:00:00:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
set protocols evpn vni-options vni 1 vrf-target export target:10003:1
set protocols evpn vni-options vni 2 vrf-target export target:10003:2
set protocols evpn vni-options vni 5 vrf-target export target:10003:5
set protocols evpn vni-options vni 6 vrf-target export target:10003:6
set protocols evpn extended-vni-list [ 1 2 5 6 ]
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 from community
  cust0001
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 from community
  cust0002
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 from community
  vni0001
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 from community
  vni0002
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 from community
  vni0005
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 from community
  vni0006
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 then accept
set policy-options community cust0001 members target:10001:1
set policy-options community cust0002 members target:10001:2
set policy-options community vni0001 members target:10003:1
set policy-options community vni0002 members target:10003:2
set policy-options community vni0005 members target:10003:5
set policy-options community vni0006 members target:10003:6
```

```
set vlans v0001 vlan-id 1
set vlans v0001 vxlan vni 1
set vlans v0001 vxlan ingress-node-replication
set vlans v0002 vlan-id 2
set vlans v0002 vxlan vni 2
set vlans v0002 vxlan ingress-node-replication
set vlans v0005 vlan-id 5
set vlans v0005 vxlan vni 5
set vlans v0005 vxlan ingress-node-replication
set vlans v0006 vlan-id 6
set vlans v0006 vxlan vni 6
set vlans v0006 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure profiles for the two customer groups:

1. Configure an aggregated Ethernet interface for the connection with the physical server. This interface is associated with VXLANs v0001, v0002, v0005, and v0006.

```
[edit interfaces]
user@switch# set xe-0/0/0 ether-options 802.3ad ae0
user@switch# set ae0 esi 00:00:00:ab:cd:00:02:00:00:01
user@switch# set ae0 esi all-active
user@switch# set ae0 aggregated-ether-options lacp active
user@switch# set ae0 aggregated-ether-options lacp system-id 00:22:00:00:00:01
user@switch# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@switch# set ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
```

2. Configure a route target for each VNI.

```
[edit protocols]
user@switch# set evpn vni-options vni 1 vrf-target export target:10003:1
user@switch# set evpn vni-options vni 2 vrf-target export target:10003:2
user@switch# set evpn vni-options vni 5 vrf-target export target:10003:5
user@switch# set evpn vni-options vni 6 vrf-target export target:10003:6
```

3. Specify which VNIs are included in the EVPN-VXLAN domains.

```
[edit protocols]
user@switch# set evpn extended-vni-list [ 1 2 5 6 ]
```

4. Configure policies that accept and import overlay routes.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 from
  community cust0001
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 from
  community cust0002
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 then accept
```

```
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 from
  community vni0001
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 from
  community vni0002
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 from
  community vni0005
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 from
  community vni0006
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 then accept
```

5. Set up communities for the customer groups and VXLANs.

```
[edit policy-options]
user@switch# set community cust0001 members target:10001:1
user@switch# set community cust0002 members target:10001:2
user@switch# set community vni0001 members target:10003:1
user@switch# set community vni0002 members target:10003:2
user@switch# set community vni0005 members target:10003:5
user@switch# set community vni0006 members target:10003:6
```

6. Configure VXLANs v0001, v0002, v0005, and v0006.

```
[edit vlans]
user@switch# set v0001 vlan-id 1
user@switch# set v0001 vxlan vni 1
user@switch# set v0001 vxlan ingress-node-replication
user@switch# set v0002 vlan-id 2
user@switch# set v0002 vxlan vni 2
user@switch# set v0002 vxlan ingress-node-replication
user@switch# set v0005 vlan-id 5
user@switch# set v0005 vxlan vni 5
user@switch# set v0005 vxlan ingress-node-replication
user@switch# set v0006 vlan-id 6
user@switch# set v0006 vxlan vni 6
user@switch# set v0006 vxlan ingress-node-replication
```

Leaf 4: Underlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces et-0/0/48 unit 0 family inet address 10.1.5.8/24
set interfaces et-0/0/50 unit 0 family inet address 10.1.29.8/24
set interfaces et-0/0/51 unit 0 family inet address 10.1.35.8/24
```

```
set interfaces et-0/0/52 unit 0 family inet address 10.1.41.8/24
set interfaces lo0 unit 0 family inet address 10.0.0.8/32 primary
set routing-options router-id 10.0.0.8
set routing-options autonomous-system 65000
set routing-options forwarding-table export load-balancing-policy
set protocols bgp multihop
set protocols bgp group underlay type external
set protocols bgp group underlay accept-remote-nexthop
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 5
set protocols bgp group underlay export send-direct
set protocols bgp group underlay peer-as 65001
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.1.5.9
set protocols bgp group underlay neighbor 10.1.29.10
set protocols bgp group underlay neighbor 10.1.35.10
set protocols bgp group underlay neighbor 10.1.41.9
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 10 from protocol direct
set policy-options policy-statement send-direct term 10 then accept
```

Configuring the Underlay Network for Leaf 4

Step-by-Step Procedure

To configure the underlay network for Leaf 4:

1. Configure Layer 3 interfaces.

   ```
   [edit interfaces]
   user@switch# set et-0/0/48 unit 0 family inet address 10.1.5.8/24
   user@switch# set et-0/0/50 unit 0 family inet address 10.1.29.8/24
   user@switch# set et-0/0/51 unit 0 family inet address 10.1.35.8/24
   user@switch# set et-0/0/52 unit 0 family inet address 10.1.41.8/24
   ```

2. Specify an IP address for the loopback interface. This IP address serves as the source IP address in the outer header of any VXLAN-encapsulated packets.

   ```
   [edit interfaces]
   user@switch# set lo0 unit 0 family inet address 10.0.0.8/32 primary
   ```

3. Set the routing options.

   ```
   [edit routing-options]
   user@switch# set router-id 10.0.0.8
   user@switch# set autonomous-system 65000
   user@switch# set forwarding-table export load-balancing-policy
   ```

4. Configure multihop EBGP.

   ```
   [edit protocols]
   ```

```
user@switch# set bgp multihop
```

5. Configure a BGP group that includes peers that handle underlay functions.

```
[edit protocols]
user@switch# set bgp group underlay type external
user@switch# set bgp group underlay accept-remote-nexthop
user@switch# set bgp group underlay advertise-peer-as
user@switch# set bgp group underlay family inet unicast loops 5
user@switch# set bgp group underlay export send-direct
user@switch# set bgp group underlay peer-as 65001
user@switch# set bgp group underlay multipath
user@switch# set bgp group underlay neighbor 10.1.5.9
user@switch# set bgp group underlay neighbor 10.1.29.10
user@switch# set bgp group underlay neighbor 10.1.35.10
user@switch# set bgp group underlay neighbor 10.1.41.9
```

NOTE: You must enter the **set protocols bgp group underlay accept-remote-nexthop** command so that the BGP routes appear in the routing table as direct routes. VXLAN requires direct next hops for equal-cost multipath (ECMP) load balancing.

6. Configure a policy that spreads traffic across multiple paths between the Juniper Networks switches.

```
[edit policy-options]
user@switch# set policy-statement load-balancing-policy then load-balance
  per-packet
```

7. Configure a policy that enables BGP to advertise direct interfaces such as loopback interface lo0.

```
[edit policy-options]
user@switch# set policy-statement send-direct term 10 from protocol direct
user@switch# set policy-statement send-direct term 10 then accept
```

Leaf 4: Overlay Network Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set protocols bgp group evpn type internal
```

```
set protocols bgp group evpn local-address 10.0.0.8
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn multipath
set protocols bgp group evpn neighbor 10.0.0.3
set protocols bgp group evpn neighbor 10.0.0.4
set protocols bgp group evpn neighbor 10.0.0.7
set protocols bgp group evpn neighbor 10.0.0.9
set protocols bgp group evpn neighbor 10.0.0.10
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
set policy-options policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
set policy-options community switch_options_comm members target:65000:2
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.0.0.8:1
set switch-options vrf-import EVPN_VRF_IMPORT
set switch-options vrf-target target:65000:2
set switch-options vrf-target auto
```

Configuring the Overlay Network for Leaf 4

Step-by-Step Procedure

To configure the overlay network for Leaf 4:

1. Configure an IBGP group for the EVPN-VXLAN overlay.

```
[edit protocols]
user@switch# set bgp group evpn type internal
user@switch# set bgp group evpn local-address 10.0.0.8
user@switch# set bgp group evpn family evpn signaling
user@switch# set bgp group evpn multipath
user@switch# set bgp group evpn neighbor 10.0.0.3
user@switch# set bgp group evpn neighbor 10.0.0.4
user@switch# set bgp group evpn neighbor 10.0.0.7
user@switch# set bgp group evpn neighbor 10.0.0.9
user@switch# set bgp group evpn neighbor 10.0.0.10
```

2. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors.

```
[edit protocols]
user@switch# set evpn encapsulation vxlan
```

3. Specify a tunnel type used for passing multicast traffic between Juniper Networks switches in an EVPN-VXLAN environment.

```
[edit protocols]
user@switch# set evpn multicast-mode ingress-replication
```

4. Set up a policy to handle Layer 2 EVPN traffic.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
  from community switch_options_comm
user@switch# set policy-statement EVPN_VRF_IMPORT term switch_options_comm
  then accept
user@switch# set community switch_options_comm members target:65000:2
```

5. Configure options for the default routing instance (virtual switch type).

```
[edit switch-options]
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 10.0.0.8:1
user@switch# set vrf-import EVPN_VRF_IMPORT
user@switch# set vrf-target target:65000:2
user@switch# set vrf-target auto
```

Leaf 4: Customer Profile Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your configuration, copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

```
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:00:00:ab:cd:00:02:00:00:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:22:00:00:00:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
set protocols evpn vni-options vni 1 vrf-target export target:10003:1
set protocols evpn vni-options vni 2 vrf-target export target:10003:2
set protocols evpn vni-options vni 5 vrf-target export target:10003:5
set protocols evpn vni-options vni 6 vrf-target export target:10003:6
set protocols evpn extended-vni-list [ 1 2 5 6 ]
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 from community
  cust0001
set policy-options policy-statement EVPN_VRF_IMPORT term cust0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 from community
  cust0002
set policy-options policy-statement EVPN_VRF_IMPORT term cust0002 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 from community
  vni0001
set policy-options policy-statement EVPN_VRF_IMPORT term vni0001 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 from community
  vni0002
set policy-options policy-statement EVPN_VRF_IMPORT term vni0002 then accept
```

```
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 from community
  vni0005
set policy-options policy-statement EVPN_VRF_IMPORT term vni0005 then accept
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 from community
  vni0006
set policy-options policy-statement EVPN_VRF_IMPORT term vni0006 then accept
set policy-options community cust0001 members target:10001:1
set policy-options community cust0002 members target:10001:2
set policy-options community vni0001 members target:10003:1
set policy-options community vni0002 members target:10003:2
set policy-options community vni0005 members target:10003:5
set policy-options community vni0006 members target:10003:6
set vlans v0001 vlan-id 1
set vlans v0001 vxlan vni 1
set vlans v0001 vxlan ingress-node-replication
set vlans v0002 vlan-id 2
set vlans v0002 vxlan vni 2
set vlans v0002 vxlan ingress-node-replication
set vlans v0005 vlan-id 5
set vlans v0005 vxlan vni 5
set vlans v0005 vxlan ingress-node-replication
set vlans v0006 vlan-id 6
set vlans v0006 vxlan vni 6
set vlans v0006 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure profiles for the two customer groups:

1. Configure an aggregated Ethernet interface for the connection with the physical server. This interface is associated with VXLANs v0001, v0002, v0005, and v0006.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/0 ether-options 802.3ad ae0
   user@switch# set ae0 esi 00:00:00:ab:cd:00:02:00:00:01
   user@switch# set ae0 esi all-active
   user@switch# set ae0 aggregated-ether-options lacp active
   user@switch# set ae0 aggregated-ether-options lacp system-id 00:22:00:00:00:01
   user@switch# set ae0 unit 0 family ethernet-switching interface-mode trunk
   user@switch# set ae0 unit 0 family ethernet-switching vlan members [ 1 2 5 6 ]
   ```

2. Configure a route target for each VNI.

   ```
   [edit protocols]
   user@switch# set evpn vni-options vni 1 vrf-target export target:10003:1
   user@switch# set evpn vni-options vni 2 vrf-target export target:10003:2
   user@switch# set evpn vni-options vni 5 vrf-target export target:10003:5
   user@switch# set evpn vni-options vni 6 vrf-target export target:10003:6
   ```

3. Specify which VNIs are included in the EVPN-VXLAN domains.

   ```
   [edit protocols]
   ```

```
user@switch# set evpn extended-vni-list [ 1 2 5 6 ]
```

4. Configure policies that accept and import overlay routes.

```
[edit policy-options]
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 from
  community cust0001
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 from
  community cust0002
user@switch# set policy-statement EVPN_VRF_IMPORT term cust0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 from
  community vni0001
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0001 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 from
  community vni0002
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0002 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 from
  community vni0005
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0005 then accept
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 from
  community vni0006
user@switch# set policy-statement EVPN_VRF_IMPORT term vni0006 then accept
```

5. Set up communities for the customer groups and VXLANs.

```
[edit policy-options]
user@switch# set community cust0001 members target:10001:1
user@switch# set community cust0002 members target:10001:2
user@switch# set community vni0001 members target:10003:1
user@switch# set community vni0002 members target:10003:2
user@switch# set community vni0005 members target:10003:5
user@switch# set community vni0006 members target:10003:6
```

6. Configure VXLANs v0001, v0002, v0005, and v0006.

```
[edit vlans]
user@switch# set v0001 vlan-id 1
user@switch# set v0001 vxlan vni 1
user@switch# set v0001 vxlan ingress-node-replication
user@switch# set v0002 vlan-id 2
user@switch# set v0002 vxlan vni 2
user@switch# set v0002 vxlan ingress-node-replication
user@switch# set v0005 vlan-id 5
user@switch# set v0005 vxlan vni 5
user@switch# set v0005 vxlan ingress-node-replication
user@switch# set v0006 vlan-id 6
user@switch# set v0006 vxlan vni 6
user@switch# set v0006 vxlan ingress-node-replication
```

Verification

Confirm that the IRB interfaces are working properly:

Verifying the Configuration of the IRB Interfaces

Purpose Verify the configuration of the IRB interfaces on Spine 1 and Spine 2.

Action From operational mode, enter the **show interfaces irb** command.

```
user@switch> show interfaces irb
```

```
Physical interface: irb    , Enabled, Physical link is Up
  Interface index: 641, SNMP ifIndex: 503
  Type: Ethernet, Link-level type: Ethernet, MTU: 1514
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
  Current address: 0c:86:10:d6:9d:fe, Hardware address: 0c:86:10:d6:9d:fe
  Last flapped   : Never
    Input packets : 0
    Output packets: 0

  Logical interface irb.1 (Index 923) (SNMP ifIndex 520)
    Flags: Up SNMP-Traps 0x4000 Encapsulation: ENET2
    Bandwidth: 1000mbps
    Routing Instance: default-switch Bridging Domain: v0001
    Input packets : 0
    Output packets: 9
    Protocol inet, MTU: 1550
      Flags: Sendbcast-pkt-to-re, Is-Primary
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.2.1/24, Local: 10.2.1.9, Broadcast: 10.2.1.255

  Logical interface irb.2 (Index 924) (SNMP ifIndex 525)
    Flags: Up SNMP-Traps 0x4000 Encapsulation: ENET2
    Bandwidth: 1000mbps
    Routing Instance: default-switch Bridging Domain: v0002
    Input packets : 0
    Output packets: 9
    Protocol inet, MTU: 1550
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.2.2/24, Local: 10.2.2.9, Broadcast: 10.2.2.255

  Logical interface irb.5 (Index 927) (SNMP ifIndex 535)
    Flags: Up SNMP-Traps 0x4000 Encapsulation: ENET2
    Bandwidth: 1000mbps
    Routing Instance: default-switch Bridging Domain: v0005
    Input packets : 0
    Output packets: 1
    Protocol inet, MTU: 1550
      Flags: Sendbcast-pkt-to-re, Is-Primary
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.2.5/24, Local: 10.2.5.9, Broadcast: 10.2.5.255

  Logical interface irb.6 (Index 928) (SNMP ifIndex 536)
    Flags: Up SNMP-Traps 0x4000 Encapsulation: ENET2
    Bandwidth: 1000mbps
    Routing Instance: default-switch Bridging Domain: v0006
    Input packets : 0
    Output packets: 1
    Protocol inet, MTU: 1550
      Flags: Sendbcast-pkt-to-re
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.2.6/24, Local: 10.2.6.9, Broadcast: 10.2.6.255
```

Meaning The sample output from Spine 2 verifies the following:

- IRB interfaces irb.1, irb.2, irb.5, and irb.6 are configured.
- The physical interface upon which the IRB interfaces are configured is up and running.
- Each IRB interface is properly mapped to its respective VXLAN.
- The configuration of each IRB interface properly reflects the IP address and destination (virtual gateway address) assigned to it.

Verifying the Configuration of the Routing Instances

Purpose Verify that the routing instances for customer groups 1 and 2 are properly configured on Spine 1 and Spine 2.

Action From operational mode, enter the **show route instance *routing-instance-name*** extensive command for customer groups 1 and 2.

user@switch> show route instance cust0001 extensive

```
cust0001:
  Router ID: 127.9.0.1
  Type: vrf               State: Active
  Interfaces:
    lo0.1
    irb.2
    irb.1
  Route-distinguisher: 10.0.0.9:2001
  Vrf-import: [ cust0001_vrf_imp ]
  Vrf-export: [ __vrf-export-cust0001-internal__ ]
  Vrf-export-target: [ target:10001:1 ]
  Fast-reroute-priority: low
  Tables:
    cust0001.inet.0        : 10 routes (10 active, 0 holddown, 0 hidden)
    cust0001.iso.0         : 0 routes (0 active, 0 holddown, 0 hidden)
    cust0001.inet6.0       : 0 routes (0 active, 0 holddown, 0 hidden)
    cust0001.mdt.0         : 0 routes (0 active, 0 holddown, 0 hidden)
```

user@switch> show route instance cust0002 extensive

```
cust0002:
  Router ID: 127.9.0.2
  Type: vrf               State: Active
  Interfaces:
    lo0.2
    irb.6
    irb.5
  Route-distinguisher: 10.0.0.9:2002
  Vrf-import: [ cust0002_vrf_imp ]
  Vrf-export: [ __vrf-export-cust0002-internal__ ]
  Vrf-export-target: [ target:10001:2 ]
  Fast-reroute-priority: low
  Tables:
    cust0002.inet.0        : 10 routes (10 active, 0 holddown, 0 hidden)
    cust0002.iso.0         : 0 routes (0 active, 0 holddown, 0 hidden)
    cust0002.inet6.0       : 0 routes (0 active, 0 holddown, 0 hidden)
    cust0002.mdt.0         : 0 routes (0 active, 0 holddown, 0 hidden)
```

Meaning In the sample output from Spine 2, the routing instances for customer groups 1 and 2 shows the loopback interface and IRB interfaces that are associated with each group. The output also shows the actual route distinguisher and import and export policy configurations.

Verifying that Dynamic MAC Addresses Are Installed

Purpose Verify that for VXLANs v0001, v0002, v0003, and v0004, a dynamic MAC address is installed in the Ethernet switching tables on Leaf 1, Leaf 2, Leaf 3, and Leaf 4.

Action From operational mode, enter the **show ethernet-switching table** command.

user@switch> 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 : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v0001               00:00:5e:00:01:01   DR       esi.7126
05:00:00:fd:e9:00:00:00:01:00
   v0001              0c:86:10:d6:9d:fe   D       vtep.32769            10.0.0.9

   v0001               dc:38:e1:6b:46:03   D        vtep.32770
10.0.0.10

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 : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v0002               00:00:5e:00:01:01   DR       esi.7127
05:00:00:fd:e9:00:00:00:02:00
   v0002              0c:86:10:d6:9d:fe   D       vtep.32769            10.0.0.9

   v0002               dc:38:e1:6b:46:03   D        vtep.32770
10.0.0.10

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 : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v0005               00:00:5e:00:01:01   DR       esi.7130
05:00:00:fd:e9:00:00:00:05:00
   v0005              0c:86:10:d6:9d:fe   D       vtep.32769            10.0.0.9
```

```
   v0005               dc:38:e1:6b:46:03   D        vtep.32770
10.0.0.10

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 : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v0006               00:00:5e:00:01:01   DR       esi.7131
05:00:00:fd:e9:00:00:00:06:00
   v0006              0c:86:10:d6:9d:fe   D       vtep.32769            10.0.0.9

   v0006               dc:38:e1:6b:46:03   D        vtep.32770
10.0.0.10
```

Meaning The sample output from Leaf 1 indicates that it has learned the MAC address 00:00:5e:00:01:01 for a virtual gateway, which is the next-hop of the ESI. The output also indicates that Leaf 1 learned the IRB MAC addresses for Spine 1 and Spine 2, which function as virtual tunnel endpoints (VTEPs).

Release History Table

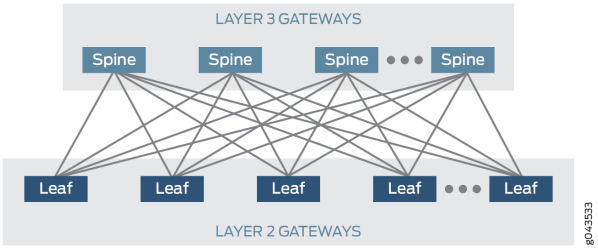
Release	Description
15.1X53-D30	Starting with Junos OS Release 15.1X53-D30, you can use integrated routing and bridging (IRB) interfaces to route data packets between Virtual Extensible LANs (VXLANs) that are implemented in an Ethernet VPN (EVPN)-VXLAN environment.

Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Topology with a Two-Layer IP Fabric

Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical [bare-metal] servers and virtual machines [VMs]) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network. Virtual Extensible LAN (VXLAN) is a tunneling protocol that creates the data plane for the Layer 2 overlay network.

The physical underlay network over which EVPN-VXLAN is commonly deployed is a two-layer IP fabric, which includes spine and leaf devices as shown in Figure 30 on page 354. In the underlay network, the spine devices provide connectivity between the leaf devices, and the leaf devices provide connectivity to the attached physical servers and VMs on virtualized servers.

Figure 30: Two-Layer IP Fabric



In an EVPN-VXLAN overlay network, the leaf devices function as Layer 2 VXLAN gateways that handle traffic within a VLAN, and the spine devices function as Layer 3 VXLAN gateways that handle traffic between VLANs using integrated routing and bridging (IRB) interfaces.

Prior to Junos OS Release 17.3R1, a QFX5110 switch can function only as a Layer 2 VXLAN gateway in a EVPN-VXLAN topology with a two-layer IP fabric, all of which is deployed within a data center. Starting with Junos OS Release 17.3R1, the QFX5110 switch can also function as a Layer 3 VXLAN gateway in an EVPN-VXLAN topology with a two-layer IP fabric.

This topic provides a sample configuration of a QFX5110 switch that functions as a spine device or Layer 3 VXLAN gateway in an EVPN-VXLAN topology with a two-layer IP fabric. This example shows how to configure Layer 3 VXLAN gateways with IRB interfaces and default gateways.

Requirements

This example uses the following hardware and software components:

- Two QFX5110 switches that function as spine devices (spine 1 and spine 2). These devices provide Layer 3 VXLAN gateway functionality.

  NOTE: This example focuses on the configuration of the QFX5110 switch that functions as spine 1. For spine 1, a basic configuration is provided for the IP/BGP underlay network, the EVPN-VXLAN overlay network, customer-specific profiles, and route leaking. This example does not include all features that can be used in an EVPN-VXLAN network. The configuration for spine 1 essentially serves as a template for the configuration of spine 2. For the configuration of spine 2, where appropriate, you can replace spine 1-specific information with the information specific to spine 2, add additional commands, and so on.

- Two QFX5200 switches that function as leaf devices (leaf 1 and leaf 2). These devices provide Layer 2 VXLAN gateway functionality.
- Junos OS Release 17.3R1 or later software running on the QFX5110 and QFX5200 switches.
- Physical servers in VLAN v100, and a physical server and a virtualized servers on which VMs are installed in VLAN v200.

Overview and Topology

In this example, a service provider supports ABC Corporation, which has multiple sites. Physical servers in site 100 must communicate with a physical servers and VMs in site 200. To enable this communication in the EVPN-VXLAN topology with the two-layer IP fabric shown in Figure 31 on page 356, on the QFX5110 switches that function as Layer 3 VXLAN gateways, or spine devices, you configure the key software entities shown in Table 16 on page 356.

Figure 31: EVPN-VXLAN Topology with Two-Layer IP Fabric

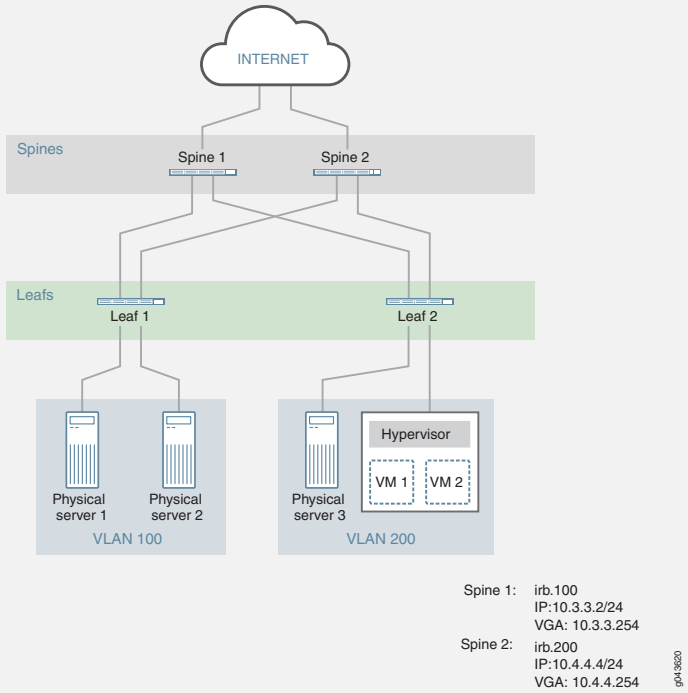


Table 16: Layer 3 Inter-VLAN Routing Entities Configured on Spine 1 and Spine 2

Entity	Configuration on Spine 1 and Spine 2
VLANs	v100 v200
VRF instances	vrf_vlan100 vrf_vlan200
IRB interfaces	irb.100 10.3.3.2/24 (IRB IP address) 10.3.3.254 (virtual gateway address)
	irb.200 10.4.4.4/24 (IRB IP address) 10.4.4.254 (virtual gateway address)

As outlined in Table 16 on page 356, on both spine devices, you configure VLAN v100 for site 100 and VLAN v200 for site 200. To segregate the Layer 3 routes associated with VLANs v100 and v200, you create VPN routing and forwarding (VRF) instances vrf_vlan100

and vrf_vlan200 on both spine devices. To route traffic between the VLANs, you configure IRB interfaces irb.100 and irb.200 on both spine devices, and associate VRF routing instance vrf_vlan100 with IRB interface irb.100, and VRF routing instance vrf_vlan200 with IRB interface irb.200.

NOTE: QFX5110 switches do not support the configuration of an IRB interface with a unique MAC address.

The physical servers in VLANs v100 and v200 are non-virtualized. As a result, we strongly recommend that you configure IRB interfaces irb.100 and irb.200 to function as default Layer 3 gateways that handle the inter-VLAN traffic of the physical servers. To that end, the configuration of each IRB interface also includes a virtual gateway address (VGA), which configures each IRB interface as a default gateway. In addition, this example assumes that each physical server is configured to use a particular default gateway. For general information about default gateways and how inter-VLAN traffic flows between a physical server to another physical server or VM in a different VLAN in an EVPN-VXLAN topology with a two-layer fabric, see "Using a Default Layer 3 Gateway to Route Traffic Between Virtual Networks in an EVPN-VXLAN Topology" on page 293.

NOTE: When configuring a VGA for an IRB interface, keep in mind that the IRB IP address and VGA must be different.

NOTE: If a QFX5110 switch running Junos OS Release 17.3R1 or later software functions as both a Layer 3 VXLAN gateway and a Dynamic Host Configuration Protocol (DHCP) relay in an EVPN-VXLAN topology, the DHCP server response time for an IP address might take up to a few minutes. The lengthy response time might occur if a DHCP client receives and later releases an IP address on an EVPN-VXLAN IRB interface configured on the QFX5110 switch and the binding between the DHCP client and the IP address is not deleted.

As outlined in Table 16 on page 356, a separate VRF routing instance is configured for each VLAN. To enable communication between the hosts in VLANs v100 and v200, this example shows how to export unicast routes from the routing table for routing instance vrf_vlan100 and import the routes into the routing table for vrf_vlan200 and vice versa. This feature is also known as route leaking.

Basic Underlay Network Configuration

CLI Quick Configuration

To quickly configure a basic underlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces et-0/0/0 unit 0 family inet address 10.1.1.1/24
```

```
set interfaces et-0/0/1 unit 0 family inet address 10.2.2.1/24
set routing-options router-id 10.2.3.11
set routing-options autonomous-system 64500
set protocols bgp group pe neighbor 10.2.3.12
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface et-0/0/0.0
set protocols ospf area 0.0.0.0 interface et-0/0/1.0
```

Configuring a Basic Underlay Network

Step-by-Step Procedure

To configure a basic underlay network on spine 1:

1. Configure the interfaces that connect to the leaf devices.

```
[edit interfaces]
user@switch# set et-0/0/0 unit 0 family inet address 10.1.1.1/24
user@switch# set et-0/0/1 unit 0 family inet address 10.2.2.1/24
```

2. Configure the router ID and autonomous system number for spine 1.

```
[edit routing-options]
user@switch# set router-id 10.2.3.11
user@switch# set autonomous-system 64500
```

3. Configure a BGP group that includes spine 2 as a peer that also handles underlay functions.

```
[edit protocols]
user@switch# set bgp group pe neighbor 10.2.3.12
```

4. Configure OSPF as the routing protocol for the underlay network.

```
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface lo0.0 passive
user@switch# set ospf area 0.0.0.0 interface et-0/0/0.0
user@switch# set ospf area 0.0.0.0 interface et-0/0/1.0
```

Basic EVPN-VXLAN Overlay Network Configuration

CLI Quick Configuration

To quickly configure a basic overlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set forwarding-options vxlan-routing interface-num 8192
set forwarding-options vxlan-routingnext-hop 16384
```

```
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.11
set protocols bgp group pe family evpn signaling
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all
set protocols evpn default-gateway no-gateway-community
set switch-options route-distinguisher 10.2.3.11:1
set switch-options vrf-target target:1111:11
set switch-options vtep-source-interface lo0.0
```

Configuring a Basic EVPN-VXLAN Overlay Network

Step-by-Step Procedure

To configure a basic EVPN-VXLAN overlay network on spine 1:

1. Increase the number of physical interfaces and next hops that the QFX5110 switch allocates for use in an EVPN-VXLAN overlay network.

   ```
   [edit forwarding-options]
   set vxlan-routing interface-num 8192
   set vxlan-routing next-hop 16384
   ```

2. Configure an IBGP overlay between spine 1 and the connected leaf devices, specify a local IP address for spine 1, and include the EVPN signaling Network Layer Reachability Information (NLRI) to the pe BGP group.

   ```
   [edit protocols]
   user@switch# set bgp group pe type internal
   user@switch# set bgp group pe local-address 10.2.3.11
   user@switch# set bgp group pe family evpn signaling
   ```

3. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors, and specify that all VXLAN network identifiers (VNIs) are part of the virtual routing and forwarding (VRF) instance. Also, specify that the MAC address of the IRB interface and the MAC address of the corresponding default gateway are advertised to the Layer 2 VXLAN gateways without the extended community option of default -gateway.

   ```
   [edit protocols]
   user@switch# set evpn encapsulation vxlan
   user@switch# set evpn extended-vni-list all
   user@switch# set evpn default-gateway no-gateway-community
   ```

4. Configure switch options to set a route distinguisher and VRF target for the VRF routing instance, and associate interface lo0 with the virtual tunnel endpoint (VTEP).

   ```
   [edit switch-options]
   user@switch# set route-distinguisher 10.2.3.11:1
   user@switch# set vrf-target target:1111:11
   user@switch# set vtep-source-interface lo0.0
   ```

Basic Customer Profile Configuration

CLI Quick Configuration

To quickly configure a basic customer profile for ABC Corporation sites 100 and 200, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces xe-0/0/32:1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/32:1 unit 0 family ethernet-switching vlan members v100
set interfaces xe-0/0/32:1 unit 0 family ethernet-switching vlan members v200
set interfaces irb unit 100 family inet address 10.3.3.2/24 virtual-gateway-address
  10.3.3.254
set interfaces irb unit 100 proxy-macip-advertisement
set interfaces irb unit 200 family inet address 10.4.4.4/24 virtual-gateway-address
  10.4.4.254
set interfaces irb unit 200 proxy-macip-advertisement
set interfaces lo0 unit 0 family inet address 10.2.3.11/32 primary
set interfaces lo0 unit 1 family inet address 10.2.3.24/32 primary
set interfaces lo0 unit 2 family inet address 10.2.3.25/32 primary
set routing-instances vrf_vlan100 instance-type vrf
set routing-instances vrf_vlan100 interface irb.100
set routing-instances vrf_vlan100 interface lo0.1
set routing-instances vrf_vlan100 route-distinguisher 10.2.3.11:2
set routing-instances vrf_vlan100 vrf-import import-inet
set routing-instances vrf_vlan100 vrf-export export-inet1
set routing-instances vrf_vlan200 instance-type vrf
set routing-instances vrf_vlan200 interface irb.200
set routing-instances vrf_vlan200 interface lo0.2
set routing-instances vrf_vlan200 route-distinguisher 10.2.3.11:3
set routing-instances vrf_vlan200 vrf-import import-inet
set routing-instances vrf_vlan200 vrf-export export-inet2
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.100
set vlans v100 vxlan vni 100
set vlans v200 vlan-id 200
set vlans v200 l3-interface irb.200
set vlans v200 vxlan vni 200
```

Configuring a Basic Customer Profile

Step-by-Step Procedure

To configure a basic customer profile for ABC Corporation sites 100 and 200 on spine 1:

1. Configure a Layer 2 interface, and specify the interface as a member of VLANs v100 and v200.

```
[edit interfaces]
user@switch# set xe-0/0/32:1 unit 0 family ethernet-switching interface-mode
  trunk
user@switch# set xe-0/0/32:1 unit 0 family ethernet-switching vlan members v100
user@switch# set xe-0/0/32:1 unit 0 family ethernet-switching vlan members v200
```

2. Create IRB interfaces, and configure the interfaces to act as default Layer 3 virtual gateways, which route traffic from physical servers in VLAN v100 to physical servers and VMs in VLAN v200 and vice versa. Also, on the IRB interfaces, enable the Layer 3 VXLAN gateway to advertise MAC+IP type 2 routes on behalf of the Layer 2 VXLAN gateways.

```
[edit interfaces]
user@switch# set irb unit 100 family inet address 10.3.3.2/24
  virtual-gateway-address 10.3.3.254
user@switch# set irb unit 100 proxy-macip-advertisement
user@switch# set irb unit 200 family inet address 10.4.4.4/24
  virtual-gateway-address 10.4.4.254
user@switch# set irb unit 200 proxy-macip-advertisement
```

NOTE: QFX5110 switches do not support the configuration of an IRB interface with a unique MAC address.

NOTE: When configuring a VGA for an IRB interface, keep in mind that the VGA and IRB IP address must be different.

3. Configure a loopback interface (lo0) for spine 1 and a logical loopback address (lo0.*x*) for each VRF routing instance.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.2.3.11/32 primary
user@switch# set lo0 unit 1 family inet address 10.2.3.24/32 primary
user@switch# set lo0 unit 2 family inet address 10.2.3.25/32 primary
```

4. Configure VRF routing instances for VLANs v100 and v200. In each routing instance, associate an IRB interface, a loopback interface, and an identifier attached to the route. Also specify that each routing instance exports its overlay routes to the VRF table for the other routing instance and imports overlay routes from the VRF table for the other routing instance into its VRF table.

```
[edit routing-instances]
user@switch# set vrf_vlan100 instance-type vrf
user@switch# set vrf_vlan100 interface irb.100
user@switch# set vrf_vlan100 interface lo0.1
user@switch# set vrf_vlan100 route-distinguisher 10.2.3.11:2
user@switch# set vrf_vlan100 vrf-import import-inet
user@switch# set vrf_vlan100 vrf-export export-inet1
user@switch# set vrf_vlan200 instance-type vrf
user@switch# set vrf_vlan200 interface irb.200
user@switch# set vrf_vlan200 interface lo0.2
user@switch# set vrf_vlan200 route-distinguisher 10.2.3.11:3
```

```
user@switch# set vrf_vlan200 vrf-import import-inet
user@switch# set vrf_vlan200 vrf-export export-inet2
```

5. Configure VLANs v100 and v200, and associate an IRB interface and VNI with each VLAN.

```
[edit vlans]
user@switch# set v100 vlan-id 100
user@switch# set v100 l3-interface irb.100
user@switch# set v100 vxlan vni 100
user@switch# set v200 vlan-id 200
user@switch# set v200 l3-interface irb.200
user@switch# set v200 vxlan vni 200
```

Route Leaking Configuration

CLI Quick Configuration

To quickly configure route leaking, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set policy-options policy-statement export-inet1 term 1 from interface irb.100
set policy-options policy-statement export-inet1 term 1 then community add com200
set policy-options policy-statement export-inet1 term 1 then accept
set policy-options policy-statement export-inet2 term 1 from interface irb.200
set policy-options policy-statement export-inet2 term 1 then community add com100
set policy-options policy-statement export-inet2 term 1 then accept
set policy-options policy-statement import-inet term 1 from community com100
set policy-options policy-statement import-inet term 1 from community com200
set policy-options policy-statement import-inet term 1 then accept
set policy-options community com100 members target:1:100
set policy-options community com200 members target:1:200
set routing-instances vrf_vlan100 vrf-import import-inet
set routing-instances vrf_vlan100 vrf-export export-inet1
set routing-instances vrf_vlan100 routing-options auto-export family inet unicast
set routing-instances vrf_vlan200 vrf-import import-inet
set routing-instances vrf_vlan200 vrf-export export-inet2
set routing-instances vrf_vlan200 routing-options auto-export family inet unicast
```

Step-by-Step Procedure

To configure route leaking on spine 1:

1. Configure a routing policy that specifies that routes learned through IRB interface irb.100 are exported and then imported into the routing table for vrf_vlan200. Configure another routing policy that specifies that routes learned through IRB interface irb.200 are exported and then imported into the routing table for vrf_vlan100.

```
[edit policy-options]
user@switch# set policy-statement export-inet1 term 1 from interface irb.100
```

```
user@switch# set policy-statement export-inet1 term 1 then community add com200
user@switch# set policy-statement export-inet1 term 1 then accept
user@switch# set policy-statement export-inet2 term 1 from interface irb.200
user@switch# set policy-statement export-inet2 term 1 then community add com100
user@switch# set policy-statement export-inet2 term 1 then accept
user@switch# set policy-statement import-inet term 1 from community com100
user@switch# set policy-statement import-inet term 1 from community com200
user@switch# set policy-statement import-inet term 1 then accept
user@switch# set community com100 members target:1:100
user@switch# set community com200 members target:1:200
```

2. In the VRF routing instances for VLANs v100 and v200. apply the routing policies configured in step 1.

```
[edit routing-instances]
user@switch# set vrf_vlan100 vrf-import import-inet
user@switch# set vrf_vlan100 vrf-export export-inet1
user@switch# set vrf_vlan200 vrf-import import-inet
user@switch# set vrf_vlan200 vrf-export export-inet2
```

3. Specify that unicast routes are to be exported from the vrf_vlan100 routing table into the vrf_vlan200 routing table and vice versa.

```
[edit routing-instances]
user@switch# set vrf_vlan100 routing-options auto-export family inet unicast
user@switch# set vrf_vlan200 routing-options auto-export family inet unicast
```

Related Documentation

- Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Topology with a Two-Layer IP Fabric on page 353

CHAPTER 10

Configuring Route Targets

Example: Configuring VNI Route Targets Automatically

This example shows how to automatically derive route targets for multiple VNIs in an EVPN-VXLAN topology.

Requirements

This example uses the following hardware and software components:

- A QFX Series switch.
- Junos OS version 15.1X53-D30

Overview

The **vrf-target** statement can be used to configure specific route targets for each VNI under **vni-options**. You can configure the **vrf-target** statement with the **auto** option to automatically derive route targets for each VNI.

Configuration

To configure the **vrf-target** statement with the **auto** option, perform these tasks:

Configuring VNI Route Target Automatic Derivation

Step-by-Step Procedure

To configure the automatic derivation of VNI route targets:

1. At the **[switch-options]** hierarchy level, configure the **vtep-source-interface**, and **route-distiguisher** statements. Then configure the **vrf-import** statement with a policy that will be configured in a later step. Next, configure the **vrf-target** statement with a target and the **auto** option. The route target configured under **vrf-target** will be used by type 1 EVPN routes. Type 2 and type 3 EVPN routes will use the auto-derived per-VNI route target for export and import.

```
[edit switch-options]
```

```
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 192.0.2.11:1
user@switch# set vrf-import import-policy
user@switch# set vrf-target target:1111:11
user@switch# set vrf-target auto
```

2. At the **[evpn]** hierarchy level, configure the **encapsulation** and **extended-vni-list** statements. For the purposes of this example, the **extended-vni-list** statement will be configured with **all**, to apply automatic route targets to all VNIs.

```
[edit protocols evpn]
```

```
user@switch# set encapsulation vxlan
user@switch# set extended-vni-list all
```

3. Configure two policies at the **[policy-statement]** hierarchy level. The first policy will be named **comglobal** and the next policy will be named **import-policy**. These policies function as an import filter that accepts routes with the auto-derived route target.

```
[edit policy-options]
```

```
user@switch# set community comglobal members target:1111:11
```

```
[edit policy-options policy-statement import-policy]
```

```
user@switch# set term 1 from community comglobal
user@switch# set term 1 then accept
user@switch# set term 100 then reject
```

Results

Use the **show** command to verify the results of your configuration.

```
user@switch> show configuration switch-options
  vtep-source-interface lo0.0;
  route-distinguisher 192.0.2.11:1;
  vrf-import imp;
  vrf-target {
      target:1111:11;
      auto;
  }
```

```
user@switch> show configuration protocols evpn
  encapsulation vxlan;
  extended-vni-list all;
```

```
user@switch> show configuration policy-options community comglobal
  members target:1111:11;
```

```
user@switch> show configuration policy-options policy-statement import-policy
  term 1{
      from community [ comglobal ];
      then accept;
  }
  term 100 {
      then reject;
  }
```

Related Documentation

- **vrf-target on page 1093**
- Example: Configuring VNI Route Targets Manually on page 367
- Example: Configuring VNI Route Targets Automatically with Manual Override on page 369

Example: Configuring VNI Route Targets Manually

This example shows how to manually set route targets for multiple VNIs in an EVPN-VXLAN topology.

- Requirements on page 367
- Overview on page 368
- Configuration on page 368

Requirements

This example uses the following hardware and software components:

- A QFX Series switch.
- Junos OS version 15.1X53-D30

Overview

The **vrf-target** statement can be used to configure specific route targets for each VNI under **vni-options**. You can configure the **vrf-target** statement to automatically derive route targets for each VNI or you can configure route targets manually. This example explains how to configure route targets manually.

Configuration

To manually configure VNI route targets, perform these tasks:

Configuring VNI Route Targets Manually

Step-by-Step Procedure

To manually configure VNI route targets:

1. At the **[switch-options]** hierarchy level, configure the **vtep-source-interface**, and **route-distiguisher** statements. Next, configure the **vrf-target** statement with a **target**. All EVPN routes for all VLANs will use the **vrf-target** address configured in this step.

```
[edit switch-options]
```

```
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 192.0.2.11:1
user@switch# set vrf-import import-policy
user@switch# set vrf-target target:1111:11
```

2. At the **[evpn]** hierarchy level, configure the **encapsulation** and **extended-vni-list** statements. For the purposes of this example, the **extended-vni-list** statement will be configured with **all**, to apply automatic route targets to all VNIs.

```
[edit protocols evpn]
```

```
user@switch# set encapsulation vxlan
user@switch# set extended-vni-list all
```

Results

After following the steps above, use the **show** command to verify the results of your configuration.

```
user@switch> show configuration switch-options
```

```
vtep-source-interface lo0.0;
route-distinguisher 192.0.2.11:1;
vrf-target {
    target:1111:11;
}
```

Related Documentation

- vrf-target on page 1093
- Example: Configuring VNI Route Targets Automatically on page 365
- Example: Configuring VNI Route Targets Automatically with Manual Override on page 369

Example: Configuring VNI Route Targets Automatically with Manual Override

This example shows how to automatically set route targets for multiple VNIs, and manually override the route target for a single VNI in an EVPN-VXLAN topology.

Requirements

This example uses the following hardware and software components:

- A QFX Series switch.
- Junos OS version 15.1X53-D30

Overview

The **vrf-target** statement can be used to configure specific route targets for each VNI under **vni-options**. You can configure the **vrf-target** statement to automatically derive route targets for each VNI or you can configure route targets manually. It's also possible to automatically derive route targets for VNIs, then manually override the route target for one or more VNIs. This example explains how to manually override the automatically assigned route targets for a specific VNI.

Configuration

To manually override an automatically configured VNI route targets, perform these tasks:

Configuring Automatic VNI Route Targets with Manual Override

Step-by-Step Procedure

To configure automatic VNI route targets with manual override:

1. At the **[switch-options]** hierarchy level, configure the **vtep-source-interface**, and **route-distiguisher** statements. Then configure the **vrf-import** statement with a policy that will be configured in a later step. Next, configure the **vrf-target** statement with a **target** and the **auto** option. The route target configured under **vrf-target** will be used by type 1 EVPN routes and all type 2 and 3 EVPN routes for all VLANs except the ones that do not match the VNI under **vni-options** in the next step.

```
[edit switch-options]
```

```
user@switch# set vtep-source-interface lo0.0
user@switch# set route-distinguisher 192.0.2.11:1
user@switch# set vrf-import import-policy
user@switch# set vrf-target target:1111:11
user@switch# set vrf-target auto
```

2. The **[evpn]** hierarchy level is where you can override the automatic assignment of VNI route targets. Configure the **vni-options** statement for VNI 100 with an export target of 1234:11. This route target will be used by type 2 and 3 EVPN routes for all VLANs that match VNI 100. Next, configure the **encapsulation** and **extended-vni-list** statements. For the purposes of this example, the **extended-vni-list** statement will be configured with only 2 VNIs.

```
[edit protocols evpn]
```

```
user@switch# vni-options vni 100 vrf-target export target:1234:11
user@switch# set encapsulation vxlan
user@switch# set extended-vni-list 100 101
```

3. Configure three policies at the **[policy-statement]** hierarchy level. The first policy will be named **comglobal**, the next policy will be named **com1234**, and the final policy will be named **import-policy**. These policies function as an import filter that accepts routes using the auto-derived route target and the manual override route target.

```
[edit policy-options policy-statement comglobal]
```

```
user@switch# set members target:1111:11
```

```
[edit policy-options policy-statement com1234]
```

```
user@switch# set members target:1234:11
```

```
[edit policy-options policy-statement import-policy]
```

```
user@switch# set term 1 from community comglobal com1234
user@switch# set term 1 then accept
user@switch# set term 100 then reject
```

Results

After following the steps above, use the **show** command to verify the results of your configuration.

```
user@switch> show configuration switch-options
  vtep-source-interface lo0.0;
  route-distinguisher 192.0.2.11:1;
  vrf-import imp;
  vrf-target {
      target:1111:11;
      auto;
  }
```

```
user@switch> show configuration protocols evpn
  vni-options {
      vni 100 {
          vrf-target export target:1234:11;
      }
  }
  encapsulation vxlan;
  extended-vni-list [ 100 101 ];
```

```
user@switch> show configuration policy-options community comglobal
  members target:1111:11;
```

```
user@switch> show configuration policy-options community com1234
  members target:1234:11;
```

```
user@switch> show configuration policy-options policy-statement import-policy
  term 1{
      from community [ com1234 comglobal ];
      then accept;
  }
  term 100 {
      then reject;
  }
```

Related Documentation

- vrf-target on page 1093

- Example: Configuring VNI Route Targets Automatically on page 365
- Example: Configuring VNI Route Targets Manually on page 367

CHAPTER 11

Configuring EVPN-VXLAN in a Topology With a Two-Layer IP Fabric

Example: Configuring an EVPN Control Plane and VXLAN Data Plane

This example shows how to configure EVPN and VXLAN on a network to support Data Center Interconnect (DCI), allow for optimal forwarding of Ethernet frames, provide network segmentation on a broad scale, enable control plane-based MAC learning, and many other advantages.

Requirements

This example uses the following hardware and software components:

- Two Juniper Networks MX Series routers to act as IP gateways for the EVPN overlay
- Four Juniper Networks QFX5100 switches. Two of these switches act as PE devices in the EVPN topology, and the other two switches act as pure IP transport for the overlay.
- Junos OS Release 16.1 or later.
- Starting with Junos OS Release 17.3R1, EVPN-VXLAN is also supported on EX9200 switches. Previously, only MPLS encapsulation was supported. In this example, the EX9200 switch would funciton as an IP gateway for the EVPN overlay. There are some configuration differences between MX Series routers and EX9200 switches, See "Configuration on EX9200 Switches" on page 147 for more information about specific EX9200 configuration.

Overview

Ethernet VPNs (EVPNs) enable you to connect groups of dispersed customer sites using Layer 2 virtual bridges, and Virtual Extensible LANs (VXLANs) enable you to stretch Layer 2 connection over an intervening Layer 3 network, while providing network segmentation

like a VLAN, but without the scaling limitation of traditional VLANs. EVPN with VXLAN encapsulation handles Layer 2 connectivity at the scale required by cloud server providers and replaces limiting protocols like STP, freeing up your Layer 3 network to use more robust routing protocols.

This example configuration shows how to configure EVPN with VXLAN encapsulation. For the purposes of this example, the MX Series routers are named Core-1 and Core-2. The QFX5100 switches are named Leaf-1, Leaf-2, Spine-1, and Spine-2. The core routers act as IP gateways for the EVPN overlay, the leaf switches act as PE devices in the EVPN topology, and the spine switches act as pure IP transport for the overlay.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Leaf-1

```
set system host-name leaf-1
set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/2 unit 0 family inet address 10.0.0.9/31
set interfaces xe-0/0/4 unit 0 family inet address 10.0.0.13/31
set interfaces xe-0/0/32 ether-options 802.3ad ae0
set interfaces xe-0/0/33 ether-options 802.3ad ae1
set interfaces xe-0/0/34 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/34 unit 0 family ethernet-switching vlan members v300
set interfaces xe-0/0/34 unit 0 family ethernet-switching vlan members v400
set interfaces ae0 esi 00:01:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode access
set interfaces ae0 unit 0 family ethernet-switching vlan members v100
set interfaces ae1 esi 00:02:02:02:02:02:02:02:02:02
set interfaces ae1 esi all-active
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp system-id 00:00:00:01:01:01
set interfaces ae1 unit 0 family ethernet-switching interface-mode access
set interfaces ae1 unit 0 family ethernet-switching vlan members v200
set interfaces lo0 unit 0 family inet address 192.0.2.2/24
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 65402
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
```

```
set protocols bgp group underlay type external
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 2
set protocols bgp group underlay export lo0
set protocols bgp group underlay peer-as 65401
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.0.0.8 description spine-1
set protocols bgp group underlay neighbor 10.0.0.12 description spine-2
set protocols bgp group EVPN_VXLAN_CORE type external
set protocols bgp group EVPN_VXLAN_CORE multihop ttl 255
set protocols bgp group EVPN_VXLAN_CORE multihop no-nexthop-change
set protocols bgp group EVPN_VXLAN_CORE local-address 192.0.2.2
set protocols bgp group EVPN_VXLAN_CORE family evpn signaling
set protocols bgp group EVPN_VXLAN_CORE peer-as 65400
set protocols bgp group EVPN_VXLAN_CORE local-as 65403
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.255.255.0 description core-1
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.255.255.1 description core-2
set protocols bgp group EVPN_VXLAN_LEAF type internal
set protocols bgp group EVPN_VXLAN_LEAF local-address 192.0.2.2
set protocols bgp group EVPN_VXLAN_LEAF family evpn signaling
set protocols bgp group EVPN_VXLAN_LEAF export LEAF-PREPEND
set protocols bgp group EVPN_VXLAN_LEAF neighbor 10.255.255.5 description leaf-2
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list 1100
set protocols evpn extended-vni-list 1200
set protocols evpn extended-vni-list 1300
set protocols evpn extended-vni-list 1400
set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-routing-options vni 1100 vrf-target export target:1:100
set protocols evpn vni-routing-options vni 1200 vrf-target export target:1:200
set protocols evpn vni-routing-options vni 1300 vrf-target export target:1:300
set protocols evpn vni-routing-options vni 1400 vrf-target export target:1:400
set protocols l2-learning traceoptions file l2ald.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set protocols lldp port-id-subtype interface-name
set protocols lldp interface all
set policy-options policy-statement LEAF-PREPEND then as-path-prepend 65402
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0
prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance
per-packet
set policy-options policy-statement vrf-imp term t1 from community com100
set policy-options policy-statement vrf-imp term t1 then accept
set policy-options policy-statement vrf-imp term t2 from community com200
set policy-options policy-statement vrf-imp term t2 then accept
set policy-options policy-statement vrf-imp term t3 from community com300
set policy-options policy-statement vrf-imp term t3 then accept
set policy-options policy-statement vrf-imp term t4 from community com400
set policy-options policy-statement vrf-imp term t4 then accept
set policy-options policy-statement vrf-imp term t5 then reject
set policy-options community com100 members target:1:100
set policy-options community com200 members target:1:200
set policy-options community com300 members target:1:300
set policy-options community com400 members target:1:400
```

```
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 192.0.2.2:1
set switch-options vrf-import vrf-imp
set switch-options vrf-target target:9999:9999
set vlans v100 vlan-id 100
set vlans v100 vxlan vni 1100
set vlans v100 vxlan ingress-node-replication
set vlans v200 vlan-id 200
set vlans v200 vxlan vni 1200
set vlans v200 vxlan ingress-node-replication
set vlans v300 vlan-id 300
set vlans v300 vxlan vni 1300
set vlans v300 vxlan ingress-node-replication
set vlans v400 vlan-id 400
set vlans v400 vxlan vni 1400
set vlans v400 vxlan ingress-node-replication
```

Leaf-2

```
set system host-name leaf-2
set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/3 unit 0 family inet address 10.0.0.11/31
set interfaces xe-0/0/5 unit 0 family inet address 10.0.0.15/31
set interfaces xe-0/0/36 ether-options 802.3ad ae0
set interfaces xe-0/0/37 ether-options 802.3ad ae1
set interfaces xe-0/0/38 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/38 unit 0 family ethernet-switching vlan members v300
set interfaces xe-0/0/38 unit 0 family ethernet-switching vlan members v100
set interfaces ae0 esi 00:01:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp passive
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode access
set interfaces ae0 unit 0 family ethernet-switching vlan members v100
set interfaces ae1 esi 00:02:02:02:02:02:02:02:02:02
set interfaces ae1 esi all-active
set interfaces ae1 aggregated-ether-options lacp passive
set interfaces ae1 aggregated-ether-options lacp system-id 00:00:00:01:01:01
set interfaces ae1 unit 0 family ethernet-switching interface-mode access
set interfaces ae1 unit 0 family ethernet-switching vlan members v200
set interfaces lo0 unit 0 family inet address 10.255.255.5/32
set routing-options router-id 10.255.255.5
set routing-options autonomous-system 65402
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 2
set protocols bgp group underlay export lo0
set protocols bgp group underlay peer-as 65401
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.0.0.10 description spine-1
set protocols bgp group underlay neighbor 10.0.0.14 description spine-2
set protocols bgp group EVPN_VXLAN_CORE type external
set protocols bgp group EVPN_VXLAN_CORE multihop ttl 255
set protocols bgp group EVPN_VXLAN_CORE multihop no-nexthop-change
set protocols bgp group EVPN_VXLAN_CORE local-address 10.255.255.5
```

```
set protocols bgp group EVPN_VXLAN_CORE family evpn signaling
set protocols bgp group EVPN_VXLAN_CORE peer-as 65400
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.255.255.0 description core-1
set protocols bgp group EVPN_VXLAN_CORE neighbor 10.255.255.1 description core-2
set protocols bgp group EVPN_VXLAN_LEAF type internal
set protocols bgp group EVPN_VXLAN_LEAF local-address 10.255.255.5
set protocols bgp group EVPN_VXLAN_LEAF family evpn signaling
set protocols bgp group EVPN_VXLAN_LEAF export LEAF-PREPEND
set protocols bgp group EVPN_VXLAN_LEAF neighbor 192.0.2.2 description leaf-1
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list 1100
set protocols evpn extended-vni-list 1200
set protocols evpn extended-vni-list 1300
set protocols evpn extended-vni-list 1400
set protocols evpn multicast-mode ingress-replication
set protocols evpn vni-routing-options vni 1100 vrf-target export target:1:100
set protocols evpn vni-routing-options vni 1200 vrf-target export target:1:200
set protocols evpn vni-routing-options vni 1300 vrf-target export target:1:300
set protocols evpn vni-routing-options vni 1400 vrf-target export target:1:400
set protocols lldp port-id-subtype interface-name
set protocols lldp interface all
set policy-options policy-statement LEAF-PREPEND then as-path-prepend 65402
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0
prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance
per-packet
set policy-options policy-statement vrf-imp term t1 from community com100
set policy-options policy-statement vrf-imp term t1 then accept
set policy-options policy-statement vrf-imp term t2 from community com200
set policy-options policy-statement vrf-imp term t2 then accept
set policy-options policy-statement vrf-imp term t3 from community com300
set policy-options policy-statement vrf-imp term t3 then accept
set policy-options policy-statement vrf-imp term t4 from community com400
set policy-options policy-statement vrf-imp term t4 then accept
set policy-options policy-statement vrf-imp term t5 then reject
set policy-options community com100 members target:1:100
set policy-options community com200 members target:1:200
set policy-options community com300 members target:1:300
set policy-options community com400 members target:1:400
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.255.255.5:1
set switch-options vrf-import vrf-imp
set switch-options vrf-target target:9999:9999
set vlans v100 vlan-id 100
set vlans v100 vxlan vni 1100
set vlans v100 vxlan ingress-node-replication
set vlans v200 vlan-id 200
set vlans v200 vxlan vni 1200
set vlans v200 vxlan ingress-node-replication
set vlans v300 vlan-id 300
set vlans v300 vxlan vni 1300
set vlans v300 vxlan ingress-node-replication
set vlans v400 vlan-id 400
set vlans v400 vxlan vni 1400
set vlans v400 vxlan ingress-node-replication
```

Spine-1

```
set system host-name spine-1
set interfaces xe-0/0/0 unit 0 family inet address 10.0.0.1/31
set interfaces xe-0/0/1 unit 0 family inet address 10.0.0.5/31
set interfaces xe-0/0/2 unit 0 family inet address 10.0.0.8/31
set interfaces xe-0/0/3 unit 0 family inet address 10.0.0.10/31
set interfaces lo0 unit 0 family inet address 10.255.255.2/32
set routing-options router-id 10.255.255.2
set routing-options autonomous-system 65401
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay-leaf type external
set protocols bgp group underlay-leaf advertise-peer-as
set protocols bgp group underlay-leaf family inet unicast loops 2
set protocols bgp group underlay-leaf export lo0
set protocols bgp group underlay-leaf peer-as 65402
set protocols bgp group underlay-leaf multipath
set protocols bgp group underlay-leaf neighbor 10.0.0.9 description leaf-1
set protocols bgp group underlay-leaf neighbor 10.0.0.11 description leaf-2
set protocols bgp group underlay-core type external
set protocols bgp group underlay-core advertise-peer-as
set protocols bgp group underlay-core family inet unicast loops 2
set protocols bgp group underlay-core export lo0
set protocols bgp group underlay-core peer-as 65400
set protocols bgp group underlay-core multipath
set protocols bgp group underlay-core neighbor 10.0.0.0 description core-1
set protocols bgp group underlay-core neighbor 10.0.0.4 description core-2
set protocols lldp port-id-subtype interface-name
set protocols lldp interface all
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0
prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance
per-packet
```

Spine-2

```
set system host-name spine-2
set interfaces xe-0/0/0 unit 0 family inet address 10.0.0.3/31
set interfaces xe-0/0/1 unit 0 family inet address 10.0.0.7/31
set interfaces xe-0/0/4 unit 0 family inet address 10.0.0.12/31
set interfaces xe-0/0/5 unit 0 family inet address 10.0.0.14/31
set interfaces lo0 unit 0 family inet address 10.255.255.3/32
set routing-options router-id 10.255.255.3
set routing-options autonomous-system 65401
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay-leaf type external
set protocols bgp group underlay-leaf advertise-peer-as
set protocols bgp group underlay-leaf family inet unicast loops 2
set protocols bgp group underlay-leaf export lo0
set protocols bgp group underlay-leaf peer-as 65402
set protocols bgp group underlay-leaf multipath
```

```
set protocols bgp group underlay-leaf neighbor 10.0.0.13 description leaf-1
set protocols bgp group underlay-leaf neighbor 10.0.0.15 description leaf-2
set protocols bgp group underlay-core type external
set protocols bgp group underlay-core advertise-peer-as
set protocols bgp group underlay-core family inet unicast loops 2
set protocols bgp group underlay-core export lo0
set protocols bgp group underlay-core peer-as 65400
set protocols bgp group underlay-core multipath
set protocols bgp group underlay-core neighbor 10.0.0.2 description core-1
set protocols bgp group underlay-core neighbor 10.0.0.6 description core-2
set protocols lldp port-id-subtype interface-name
set protocols lldp interface all
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0
prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance
per-packet
```

Core-1

```
set system host-name core-1
set interfaces ge-1/0/0 unit 0 family inet address 10.0.0.1/31
set interfaces ge-1/0/1 unit 0 family inet address 10.0.0.2/31
set interfaces irb unit 1100 family inet address 172.16.0.2/24
virtual-gateway-address 100.0.0.1
set interfaces irb unit 1200 family inet address 172.16.0.3/24
virtual-gateway-address 200.0.0.1
set interfaces irb unit 1300 family inet address 10.10.10.2/24
virtual-gateway-address 10.10.10.1
set interfaces irb unit 1400 family inet address 10.10.10.2/24
virtual-gateway-address 10.10.10.1
set interfaces lo0 unit 0 family inet address 10.255.255.2/32
set routing-options router-id 10.255.255.2
set routing-options autonomous-system 65400
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 2
set protocols bgp group underlay export lo0
set protocols bgp group underlay peer-as 65401
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.0.0.1 description spine-1
set protocols bgp group underlay neighbor 10.0.0.3 description spine-2
set protocols bgp group EVPN_VXLAN type external
set protocols bgp group EVPN_VXLAN local-address 10.255.255.2
set protocols bgp group EVPN_VXLAN family evpn signaling
set protocols bgp group EVPN_VXLAN peer-as 65402
set protocols bgp group EVPN_VXLAN multipath
set protocols bgp group EVPN_VXLAN neighbor 192.0.2.2 description leaf-1
set protocols bgp group EVPN_VXLAN neighbor 192.0.2.2 multihop ttl 255
set protocols bgp group EVPN_VXLAN neighbor 10.255.255.5 description leaf-2
set protocols bgp group EVPN_VXLAN neighbor 10.255.255.5 multihop ttl 255
set protocols bgp group EVPN_VXLAN_TEMP type external
set protocols bgp group EVPN_VXLAN_TEMP local-address 10.255.255.2
```

```
set protocols bgp group EVPN_VXLAN_TEMP family evpn signaling
set protocols bgp group EVPN_VXLAN_TEMP peer-as 65403
set protocols bgp group EVPN_VXLAN_TEMP multipath
set protocols bgp group EVPN_VXLAN_TEMP neighbor 192.0.2.2 description leaf-1
set protocols bgp group EVPN_VXLAN_TEMP neighbor 192.0.2.2 multihop ttl 255
set protocols bgp group EVPN_VXLAN_TEMP neighbor 10.255.255.5 description leaf-2
set protocols bgp group EVPN_VXLAN_TEMP neighbor 10.255.255.5 multihop ttl 255
set protocols lldp port-id-subtype interface-name
set protocols lldp interface all
set policy-options policy-statement VS_VLAN100_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN100_IMP term ESI then accept
set policy-options policy-statement VS_VLAN100_IMP term VS_VLAN100 from community
 comm-VS_VLAN100
set policy-options policy-statement VS_VLAN100_IMP term VS_VLAN100 then accept
set policy-options policy-statement VS_VLAN200_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN200_IMP term ESI then accept
set policy-options policy-statement VS_VLAN200_IMP term VS_VLAN200 from community
 comm-VS_VLAN200
set policy-options policy-statement VS_VLAN200_IMP term VS_VLAN200 then accept
set policy-options policy-statement VS_VLAN300_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN300_IMP term ESI then accept
set policy-options policy-statement VS_VLAN300_IMP term VS_VLAN300 from community
 comm-VS_VLAN300
set policy-options policy-statement VS_VLAN300_IMP term VS_VLAN300 then accept
set policy-options policy-statement VS_VLAN400_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN400_IMP term ESI then accept
set policy-options policy-statement VS_VLAN400_IMP term VS_VLAN400 from community
 comm-VS_VLAN400
set policy-options policy-statement VS_VLAN400_IMP term VS_VLAN400 then accept
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0
prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance
per-packet
set policy-options community comm-VS_VLAN100 members target:1:100
set policy-options community comm-VS_VLAN200 members target:1:200
set policy-options community comm-VS_VLAN300 members target:1:300
set policy-options community comm-VS_VLAN400 members target:1:400
set policy-options community comm-leaf_esi members target:9999:9999
set routing-instances VRF_Tenant_A instance-type vrf
set routing-instances VRF_Tenant_A interface irb.1100
set routing-instances VRF_Tenant_A route-distinguisher 10.255.255.2:1100
set routing-instances VRF_Tenant_A vrf-target target:10:100
set routing-instances VRF_Tenant_A routing-options auto-export
set routing-instances VRF_Tenant_B instance-type vrf
set routing-instances VRF_Tenant_B interface irb.1200
set routing-instances VRF_Tenant_B route-distinguisher 10.255.255.2:1200
set routing-instances VRF_Tenant_B vrf-target target:10:200
set routing-instances VRF_Tenant_C instance-type vrf
set routing-instances VRF_Tenant_C interface irb.1300
set routing-instances VRF_Tenant_C route-distinguisher 10.255.255.2:1300
set routing-instances VRF_Tenant_C vrf-target target:10:300
set routing-instances VRF_Tenant_D instance-type vrf
```

```
set routing-instances VRF_Tenant_D interface irb.1400
set routing-instances VRF_Tenant_D route-distinguisher 10.255.255.2:1400
set routing-instances VRF_Tenant_D vrf-target target:10:400
set routing-instances VS_VLAN100 vtep-source-interface lo0.0
set routing-instances VS_VLAN100 instance-type virtual-switch
set routing-instances VS_VLAN100 route-distinguisher 10.255.255.2:100
set routing-instances VS_VLAN100 vrf-import VS_VLAN100_IMP
set routing-instances VS_VLAN100 vrf-target target:1:100
set routing-instances VS_VLAN100 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN100 protocols evpn extended-vni-list 1100
set routing-instances VS_VLAN100 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN100 bridge-domains bd1100 vlan-id 100
set routing-instances VS_VLAN100 bridge-domains bd1100 routing-interface irb.1100
set routing-instances VS_VLAN100 bridge-domains bd1100 vxlan vni 1100
set routing-instances VS_VLAN100 bridge-domains bd1100 vxlan
ingress-node-replication
set routing-instances VS_VLAN200 vtep-source-interface lo0.0
set routing-instances VS_VLAN200 instance-type virtual-switch
set routing-instances VS_VLAN200 route-distinguisher 10.255.255.2:200
set routing-instances VS_VLAN200 vrf-import VS_VLAN200_IMP
set routing-instances VS_VLAN200 vrf-target target:1:200
set routing-instances VS_VLAN200 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN200 protocols evpn extended-vni-list 1200
set routing-instances VS_VLAN200 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN200 bridge-domains bd1200 vlan-id 200
set routing-instances VS_VLAN200 bridge-domains bd1200 routing-interface irb.1200
set routing-instances VS_VLAN200 bridge-domains bd1200 vxlan vni 1200
set routing-instances VS_VLAN200 bridge-domains bd1200 vxlan
ingress-node-replication
set routing-instances VS_VLAN300 vtep-source-interface lo0.0
set routing-instances VS_VLAN300 instance-type virtual-switch
set routing-instances VS_VLAN300 route-distinguisher 10.255.255.2:300
set routing-instances VS_VLAN300 vrf-import VS_VLAN300_IMP
set routing-instances VS_VLAN300 vrf-target target:1:300
set routing-instances VS_VLAN300 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN300 protocols evpn extended-vni-list 1300
set routing-instances VS_VLAN300 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN300 bridge-domains bd1300 vlan-id 300
set routing-instances VS_VLAN300 bridge-domains bd1300 routing-interface irb.1300
set routing-instances VS_VLAN300 bridge-domains bd1300 vxlan vni 1300
set routing-instances VS_VLAN300 bridge-domains bd1300 vxlan
ingress-node-replication
set routing-instances VS_VLAN400 vtep-source-interface lo0.0
set routing-instances VS_VLAN400 instance-type virtual-switch
set routing-instances VS_VLAN400 route-distinguisher 10.255.255.2:400
set routing-instances VS_VLAN400 vrf-import VS_VLAN400_IMP
set routing-instances VS_VLAN400 vrf-target target:1:400
set routing-instances VS_VLAN400 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN400 protocols evpn extended-vni-list 1400
set routing-instances VS_VLAN400 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN400 bridge-domains bd1400 vlan-id 400
set routing-instances VS_VLAN400 bridge-domains bd1400 routing-interface irb.1400
set routing-instances VS_VLAN400 bridge-domains bd1400 vxlan vni 1400
set routing-instances VS_VLAN400 bridge-domains bd1400 vxlan
ingress-node-replication
```

EX9200 Configuration

On EX9200 switches, the **vlans** statement is used instead of **bridge-domains**, and the **l3-interface** statement is used instead of **routing-interface**.

The following example shows how to configure these statements. All other configuration shown for MX Series routers in this example applies to EX9200 switches.

```
set routing-instances VS_VLAN300 vlans vlan1300 vlan-id 300
set routing-instances VS_VLAN300 vlans vlan1300 l3-inteface irb.1300
```

NOTE: In this example, wherever **bridge-domains** or **routing-interface** statements are used, to configure on EX9200 switches, use **vlans** and **l3-interface** instead.

Core-2

```
set system host-name core-2
set chassis network-services enhanced-ip
set interfaces ge-1/0/0 unit 0 family inet address 10.0.0.4/31
set interfaces ge-1/0/1 unit 0 family inet address 10.0.0.6/31
set interfaces irb unit 1100 family inet address 172.16.0.4/24
virtual-gateway-address 100.0.0.1
set interfaces irb unit 1200 family inet address 172.16.0.5/24
virtual-gateway-address 200.0.0.1
set interfaces irb unit 1300 family inet address 10.10.10.3/24
virtual-gateway-address 10.10.10.1
set interfaces irb unit 1400 family inet address 10.10.10.3/24
virtual-gateway-address 10.10.10.1
set interfaces lo0 unit 0 family inet address 10.255.255.1/32
set routing-options router-id 10.255.255.1
set routing-options autonomous-system 65400
set routing-options forwarding-table export load-balance
set routing-options forwarding-table ecmp-fast-reroute
set protocols bgp group underlay type external
set protocols bgp group underlay advertise-peer-as
set protocols bgp group underlay family inet unicast loops 2
set protocols bgp group underlay export lo0
set protocols bgp group underlay peer-as 65401
set protocols bgp group underlay multipath
set protocols bgp group underlay neighbor 10.0.0.5 description spine-1
set protocols bgp group underlay neighbor 10.0.0.7 description spine-2
set protocols bgp group EVPN_VXLAN type external
set protocols bgp group EVPN_VXLAN local-address 10.255.255.1
set protocols bgp group EVPN_VXLAN family evpn signaling
set protocols bgp group EVPN_VXLAN peer-as 65402
set protocols bgp group EVPN_VXLAN multipath
set protocols bgp group EVPN_VXLAN neighbor 192.0.2.2 description leaf-1
set protocols bgp group EVPN_VXLAN neighbor 192.0.2.2 multihop ttl 255
set protocols bgp group EVPN_VXLAN neighbor 10.255.255.5 description leaf-2
set protocols bgp group EVPN_VXLAN neighbor 10.255.255.5 multihop ttl 255
```

```
set protocols lldp port-id-subtype interface-name
set protocols lldp interface all
set policy-options policy-statement VS_VLAN100_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN100_IMP term ESI then accept
set policy-options policy-statement VS_VLAN100_IMP term VS_VLAN100 from community
 comm-VS_VLAN100
set policy-options policy-statement VS_VLAN100_IMP term VS_VLAN100 then accept
set policy-options policy-statement VS_VLAN200_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN200_IMP term ESI then accept
set policy-options policy-statement VS_VLAN200_IMP term VS_VLAN200 from community
 comm-VS_VLAN200
set policy-options policy-statement VS_VLAN200_IMP term VS_VLAN200 then accept
set policy-options policy-statement VS_VLAN300_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN300_IMP term ESI then accept
set policy-options policy-statement VS_VLAN300_IMP term VS_VLAN300 from community
 comm-VS_VLAN300
set policy-options policy-statement VS_VLAN300_IMP term VS_VLAN300 then accept
set policy-options policy-statement VS_VLAN400_IMP term ESI from community
comm-leaf_esi
set policy-options policy-statement VS_VLAN400_IMP term ESI then accept
set policy-options policy-statement VS_VLAN400_IMP term VS_VLAN400 from community
 comm-VS_VLAN400
set policy-options policy-statement VS_VLAN400_IMP term VS_VLAN400 then accept
set policy-options policy-statement lo0 from family inet
set policy-options policy-statement lo0 from protocol direct
set policy-options policy-statement lo0 from route-filter 0.0.0.0/0
prefix-length-range /32-/32
set policy-options policy-statement lo0 then accept
set policy-options policy-statement load-balance term 1 then load-balance
per-packet
set policy-options community comm-VS_VLAN100 members target:1:100
set policy-options community comm-VS_VLAN200 members target:1:200
set policy-options community comm-VS_VLAN300 members target:1:300
set policy-options community comm-VS_VLAN400 members target:1:400
set policy-options community comm-leaf_esi members target:9999:9999
set routing-instances VRF_Tenant_A instance-type vrf
set routing-instances VRF_Tenant_A interface irb.1100
set routing-instances VRF_Tenant_A route-distinguisher 10.255.255.1:1100
set routing-instances VRF_Tenant_A vrf-target target:10:100
set routing-instances VRF_Tenant_A routing-options auto-export
set routing-instances VRF_Tenant_B instance-type vrf
set routing-instances VRF_Tenant_B interface irb.1200
set routing-instances VRF_Tenant_B route-distinguisher 10.255.255.1:1200
set routing-instances VRF_Tenant_B vrf-target target:10:200
set routing-instances VRF_Tenant_C instance-type vrf
set routing-instances VRF_Tenant_C interface irb.1300
set routing-instances VRF_Tenant_C route-distinguisher 10.255.255.1:1300
set routing-instances VRF_Tenant_C vrf-target target:10:300
set routing-instances VRF_Tenant_D instance-type vrf
set routing-instances VRF_Tenant_D interface irb.1400
set routing-instances VRF_Tenant_D route-distinguisher 10.255.255.1:1400
set routing-instances VRF_Tenant_D vrf-target target:10:400
set routing-instances VS_VLAN100 vtep-source-interface lo0.0
set routing-instances VS_VLAN100 instance-type virtual-switch
set routing-instances VS_VLAN100 route-distinguisher 10.255.255.1:100
set routing-instances VS_VLAN100 vrf-import VS_VLAN100_IMP
```

```
set routing-instances VS_VLAN100 vrf-target target:1:100
set routing-instances VS_VLAN100 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN100 protocols evpn extended-vni-list 1100
set routing-instances VS_VLAN100 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN100 protocols evpn default-gateway
no-gateway-community
set routing-instances VS_VLAN100 bridge-domains bd1100 vlan-id 100
set routing-instances VS_VLAN100 bridge-domains bd1100 routing-interface irb.1100
set routing-instances VS_VLAN100 bridge-domains bd1100 vxlan vni 1100
set routing-instances VS_VLAN100 bridge-domains bd1100 vxlan
ingress-node-replication
set routing-instances VS_VLAN200 vtep-source-interface lo0.0
set routing-instances VS_VLAN200 instance-type virtual-switch
set routing-instances VS_VLAN200 route-distinguisher 10.255.255.1:200
set routing-instances VS_VLAN200 vrf-import VS_VLAN200_IMP
set routing-instances VS_VLAN200 vrf-target target:1:200
set routing-instances VS_VLAN200 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN200 protocols evpn extended-vni-list 1200
set routing-instances VS_VLAN200 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN200 bridge-domains bd1200 vlan-id 200
set routing-instances VS_VLAN200 bridge-domains bd1200 routing-interface irb.1200
set routing-instances VS_VLAN200 bridge-domains bd1200 vxlan vni 1200
set routing-instances VS_VLAN200 bridge-domains bd1200 vxlan
ingress-node-replication
set routing-instances VS_VLAN300 vtep-source-interface lo0.0
set routing-instances VS_VLAN300 instance-type virtual-switch
set routing-instances VS_VLAN300 route-distinguisher 10.255.255.1:300
set routing-instances VS_VLAN300 vrf-import VS_VLAN300_IMP
set routing-instances VS_VLAN300 vrf-target target:1:300
set routing-instances VS_VLAN300 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN300 protocols evpn extended-vni-list 1300
set routing-instances VS_VLAN300 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN300 bridge-domains bd1300 vlan-id 300
set routing-instances VS_VLAN300 bridge-domains bd1300 routing-interface irb.1300
set routing-instances VS_VLAN300 bridge-domains bd1300 vxlan vni 1300
set routing-instances VS_VLAN300 bridge-domains bd1300 vxlan
ingress-node-replication
set routing-instances VS_VLAN400 vtep-source-interface lo0.0
set routing-instances VS_VLAN400 instance-type virtual-switch
set routing-instances VS_VLAN400 route-distinguisher 10.255.255.1:400
set routing-instances VS_VLAN400 vrf-import VS_VLAN400_IMP
set routing-instances VS_VLAN400 vrf-target target:1:400
set routing-instances VS_VLAN400 protocols evpn encapsulation vxlan
set routing-instances VS_VLAN400 protocols evpn extended-vni-list 1400
set routing-instances VS_VLAN400 protocols evpn multicast-mode ingress-replication
set routing-instances VS_VLAN400 bridge-domains bd1400 vlan-id 400
set routing-instances VS_VLAN400 bridge-domains bd1400 routing-interface irb.1400
set routing-instances VS_VLAN400 bridge-domains bd1400 vxlan vni 1400
set routing-instances VS_VLAN400 bridge-domains bd1400 vxlan
ingress-node-replication
```

Configuring Leaf-1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the QFX5100 switch called Leaf-1:

1. Set the system hostname.

```
[edit]
```

```
user@leaf-1# set system host-name leaf-1
```

2. Configure routing options.

```
[edit routing-options]
```

```
user@leaf-1# set router-id 192.0.2.2
user@leaf-1# set autonomous-system 65402
user@leaf-1# set forwarding-table export load-balance
user@leaf-1# set forwarding-table ecmp-fast-reroute
```

3. Configure load balancing.

```
[edit policy-options policy-statement load-balance]
```

```
user@leaf-1# set term 1 then load-balance per-packet
```

4. Make sure lo0 is exported and thus advertised into the underlay, then configure **family inet unicast loops 2**. Doing this is necessary because of the design choice to re-use the same autonomous system (AS) number within a tier.

```
[edit protocols bgp group underlay]
```

```
user@leaf-1# set type external
user@leaf-1# set advertise-peer-as
user@leaf-1# set family inet unicast loops 2
user@leaf-1# set export lo0
user@leaf-1# set peer-as 65401
user@leaf-1# set multipath
```

```
user@leaf-1# set neighbor 10.0.0.8 description spine-1
user@leaf-1# set neighbor 10.0.0.12 description spine-2
```

5. Configure policy options for the loopback address.

```
[edit policy-options policy-statement lo0]
```

```
user@leaf-1# set from family inet
user@leaf-1# set from protocol direct
user@leaf-1# set from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@leaf-1# set then accept
```

6. Configure switch options. At the **[switch-options]** hierarchy level, the virtual tunnel endpoint interface is lo0.0, which must be reachable through the underlay routing protocol. The route distinguisher must be unique across all switches in the network to ensure all route advertisements within MP-BGP are globally unique. The VRF table target on the QFX Series switch is, at a minimum, the community with which the switch sends all ESI (Type-1) routes. The **vrf-import vrf-imp** statement defines the target community list, which is imported into the default-switch.evpn.0 instance from bgp.evpn.0.

```
[edit switch-options]
```

```
user@leaf-1# set vtep-source-interface lo0.0
user@leaf-1# set route-distinguisher 192.0.2.2:1
user@leaf-1# set vrf-import vrf-imp
user@leaf-1# set vrf-target target:9999:9999
```

7. Configure the VRF table import policy.

```
[edit policy-options policy-statement vrf-imp]
```

```
user@leaf-1# set term t1 from community com100
user@leaf-1# set vrf-imp term t1 then accept
user@leaf-1# set vrf-imp term t2 from community com200
user@leaf-1# set vrf-imp term t2 then accept
user@leaf-1# set vrf-imp term t3 from community com300
user@leaf-1# set vrf-imp term t3 then accept
user@leaf-1# set vrf-imp term t4 from community com400
user@leaf-1# set vrf-imp term t4 then accept
user@leaf-1# set vrf-imp term t5 then reject
```

8. Configure the community policy options.

```
[edit policy-options community]
```

```
user@leaf-1# set com100 members target:1:100
user@leaf-1# set com200 members target:1:200
user@leaf-1# set com300 members target:1:300
user@leaf-1# set com400 members target:1:400
```

9. Configure the extended virtual network identifier list to establish which VXLAN network identifiers you want to be part of the EVPN and VXLAN MP-BGP domain. Next, set up the ingress replication; this EVPN and VXLAN ingress-replication is used instead of a multicast underlay. Then configure different route targets for each VXLAN network identifier instance under **vni-routing-options**.

```
[edit protocols evpn]
```

```
user@leaf-1# set encapsulation vxlan
user@leaf-1# set extended-vni-list 1100
user@leaf-1# set extended-vni-list 1200
user@leaf-1# set extended-vni-list 1300
user@leaf-1# set extended-vni-list 1400
user@leaf-1# set multicast-mode ingress-replication
user@leaf-1# set vni-routing-options vni 1100 vrf-target export target:1:100
user@leaf-1# set vni-routing-options vni 1200 vrf-target export target:1:200
user@leaf-1# set vni-routing-options vni 1300 vrf-target export target:1:300
user@leaf-1# set vni-routing-options vni 1400 vrf-target export target:1:400
```

10. Map locally significant VLAN IDs to globally significant VXLAN network identifiers.

```
[edit vlans]
```

```
user@leaf-1# set v100 vlan-id 100
user@leaf-1# set v100 vxlan vni 1100
user@leaf-1# set v100 vxlan ingress-node-replication
user@leaf-1# set v200 vlan-id 200
user@leaf-1# set v200 vxlan vni 1200
user@leaf-1# set v200 vxlan ingress-node-replication
user@leaf-1# set v300 vlan-id 300
user@leaf-1# set v300 vxlan vni 1300
user@leaf-1# set v300 vxlan ingress-node-replication
user@leaf-1# set v400 vlan-id 400
user@leaf-1# set v400 vxlan vni 1400
user@leaf-1# set v400 vxlan ingress-node-replication
```

11. Configure the EVPN MP-BGP sessions.

```
[edit protocols bgp group EVPN_VXLAN_CORE]
```

```
user@leaf-1# set type external
user@leaf-1# set multihop ttl 255
user@leaf-1# set multihop no-nexthop-change
user@leaf-1# set local-address 192.0.2.2
user@leaf-1# set family evpn signaling
user@leaf-1# set peer-as 65400
user@leaf-1# set local-as 65403
user@leaf-1# set neighbor 10.255.255.0 description core-1
user@leaf-1# set neighbor 10.255.255.1 description core-2
user@leaf-1# set type internal
user@leaf-1# set local-address 192.0.2.2
user@leaf-1# set family evpn signaling
user@leaf-1# set export LEAF-PREPEND
user@leaf-1# set neighbor 10.255.255.5 description leaf-2
```

12. Configure the Gigabit Ethernet interfaces.

```
[edit interfaces]
```

```
user@leaf-1# set xe-0/0/2 unit 0 family inet address 10.0.0.9/31
user@leaf-1# set xe-0/0/4 unit 0 family inet address 10.0.0.13/31
user@leaf-1# set xe-0/0/32 ether-options 802.3ad ae0
user@leaf-1# set xe-0/0/33 ether-options 802.3ad ae1
user@leaf-1# set xe-0/0/34 unit 0 family ethernet-switching interface-mode
trunk
user@leaf-1# set xe-0/0/34 unit 0 family ethernet-switching vlan members
v300
user@leaf-1# set xe-0/0/34 unit 0 family ethernet-switching vlan members
v400
```

13. Configure two LACP-enabled LAG interfaces. The ESI value is globally unique across the entire EVPN domain. The **all-active** configuration statement ensures that all PE routers to which this multihomed tenant is attached to can forward traffic from the CE device, such that all CE links are actively used.

```
[edit interfaces]
```

```
user@leaf-1# set ae0 esi 00:01:01:01:01:01:01:01:01:01
user@leaf-1# set ae0 esi all-active
user@leaf-1# set e0 aggregated-ether-options lacp active
user@leaf-1# set ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
user@leaf-1# set ae0 unit 0 family ethernet-switching interface-mode access
user@leaf-1# set ae0 unit 0 family ethernet-switching vlan members v100
user@leaf-1# set ae1 esi 00:02:02:02:02:02:02:02:02:02
user@leaf-1# set ae1 esi all-active
user@leaf-1# set ae1 aggregated-ether-options lacp active
user@leaf-1# set ae1 aggregated-ether-options lacp system-id 00:00:00:01:01:01
```

```
user@leaf-1# set ae1 unit 0 family ethernet-switching interface-mode access
user@leaf-1# set ae1 unit 0 family ethernet-switching vlan members v200
```

14. Configure the loopback interface address.

```
[edit interfaces]
```

```
user@leaf-1# set lo0 unit 0 family inet address 192.0.2.2/24
```

Configuring Leaf-2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

The configuration of Leaf-2 is very similar to the configuration of Leaf-1. Configure the QFX5100 switch called Leaf-2:

1. Set the system hostname.

```
[edit]
```

```
user@leaf-2# set system host-name leaf-2
```

2. Configure routing options.

```
[edit routing-options]
```

```
user@leaf-2# set router-id 10.255.255.5
user@leaf-2# set autonomous-system 65402
user@leaf-2# set forwarding-table export load-balance
user@leaf-2# set forwarding-table ecmp-fast-reroute
```

3. Configure load balancing.

```
[edit policy-options policy-statement load-balance]
```

```
user@leaf-2# set term 1 then load-balance per-packet
```

4. Make sure lo0 is exported and thus advertised into the underlay, then configure **family inet unicast loops 2**. Doing this is necessary because of the design choice to re-use the same autonomous system (AS) number within a tier.

```
[edit protocols bgp group underlay]
```

```
user@leaf-2# set type external
user@leaf-2# set advertise-peer-as
user@leaf-2# set family inet unicast loops 2
user@leaf-2# set export lo0
user@leaf-2# set peer-as 65401
user@leaf-2# set multipath
user@leaf-2# set neighbor 10.0.0.10 description spine-1
user@leaf-2# set neighbor 10.0.0.14 description spine-2
```

5. Configure policy options for the loopback address.

```
[edit policy-options policy-statement lo0]
```

```
user@leaf-2# set from family inet
user@leaf-2# set from protocol direct
user@leaf-2# set from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@leaf-2# set then accept
```

6. Configure switch options. At the **[switch-options]** hierarchy level, the virtual tunnel endpoint interface is lo0.0, which must be reachable through the underlay routing protocol. The route distinguisher must be unique across all switches in the network to ensure all route advertisements within MP-BGP are globally unique. The VRF table target on the QFX Series switch is, at a minimum, the community with which the switch sends all ESI (Type-1) routes. The **vrf-import vrf-imp** statement defines the target community list, which is imported into the default-switch.evpn.0 instance from bgp.evpn.0.

```
[edit switch-options]
```

```
user@leaf-2# set vtep-source-interface lo0.0
user@leaf-2# set route-distinguisher 10.255.255.5:1
user@leaf-2# set vrf-import vrf-imp
user@leaf-2# set vrf-target target:9999:9999
```

7. Configure the VRF table import policy.

```
[edit policy-options policy-statement vrf-imp]
```

```
user@leaf-2# set term t1 from community com100
user@leaf-2# set vrf-imp term t1 then accept
user@leaf-2# set vrf-imp term t2 from community com200
user@leaf-2# set vrf-imp term t2 then accept
user@leaf-2# set vrf-imp term t3 from community com300
user@leaf-2# set vrf-imp term t3 then accept
user@leaf-2# set vrf-imp term t4 from community com400
user@leaf-2# set vrf-imp term t4 then accept
user@leaf-2# set vrf-imp term t5 then reject
```

8. Configure the community policy options.

```
[edit policy-options community]
```

```
user@leaf-2# set com100 members target:1:100
user@leaf-2# set com200 members target:1:200
user@leaf-2# set com300 members target:1:300
user@leaf-2# set com400 members target:1:400
```

9. Configure the extended virtual network identifier list to establish which VXLAN network identifiers you want to be part of the EVPN and VXLAN MP-BGP domain. Next, set up the ingress replication; this EVPN and VXLAN ingress-replication is used instead of a multicast underlay. Then configure different route targets for each VXLAN network identifier instance under **vni-routing-options**.

```
[edit protocols evpn]
```

```
user@leaf-2# set encapsulation vxlan
user@leaf-2# set extended-vni-list 1100
user@leaf-2# set extended-vni-list 1200
user@leaf-2# set extended-vni-list 1300
user@leaf-2# set extended-vni-list 1400
user@leaf-2# set multicast-mode ingress-replication
user@leaf-2# set vni-routing-options vni 1100 vrf-target export target:1:100
user@leaf-2# set vni-routing-options vni 1200 vrf-target export target:1:200
user@leaf-2# set vni-routing-options vni 1300 vrf-target export target:1:300
user@leaf-2# set vni-routing-options vni 1400 vrf-target export target:1:400
```

10. Map locally significant VLAN IDs to globally significant VXLAN network identifiers.

```
[edit vlans]
```

```
user@leaf-2# set v100 vlan-id 100
user@leaf-2# set v100 vxlan vni 1100
user@leaf-2# set v100 vxlan ingress-node-replication
user@leaf-2# set v200 vlan-id 200
```

```
user@leaf-2# set v200 vxlan vni 1200
user@leaf-2# set v200 vxlan ingress-node-replication
user@leaf-2# set v300 vlan-id 300
user@leaf-2# set v300 vxlan vni 1300
user@leaf-2# set v300 vxlan ingress-node-replication
user@leaf-2# set v400 vlan-id 400
user@leaf-2# set v400 vxlan vni 1400
user@leaf-2# set v400 vxlan ingress-node-replication
```

11. Configure the EVPN MP-BGP sessions.

```
[edit protocols bgp group EVPN_VXLAN_CORE]
```

```
user@leaf-2# set type external
user@leaf-2# set multihop ttl 255
user@leaf-2# set multihop no-nexthop-change
user@leaf-2# set local-address 10.255.255.5
user@leaf-2# set family evpn signaling
user@leaf-2# set peer-as 65400
user@leaf-2# set neighbor 10.255.255.2 description core-1
user@leaf-2# set neighbor 10.255.255.1 description core-2
user@leaf-2# set type internal
user@leaf-2# set local-address 10.255.255.5
user@leaf-2# set family evpn signaling
user@leaf-2# set export LEAF-PREPEND
user@leaf-2# set neighbor 192.0.2.2 description leaf-2
```

12. Configure the Gigabit Ethernet interfaces.

```
[edit interfaces]
```

```
user@leaf-2# set xe-0/0/3 unit 0 family inet address 10.0.0.11/31
user@leaf-2# set xe-0/0/5 unit 0 family inet address 10.0.0.15/31
user@leaf-2# set xe-0/0/36 ether-options 802.3ad ae0
user@leaf-2# set xe-0/0/37 ether-options 802.3ad ae1
user@leaf-2# set xe-0/0/38 unit 0 family ethernet-switching interface-mode
trunk
user@leaf-2# set xe-0/0/38 unit 0 family ethernet-switching vlan members
v300
user@leaf-2# set xe-0/0/38 unit 0 family ethernet-switching vlan members
v100
```

13. Configure two LACP-enabled LAG interfaces. The ESI value is globally unique across the entire EVPN domain. The **all-active** configuration statement ensures that all PE routers to which this multihomed tenant is attached to can forward traffic from the CE device, such that all CE links are actively used.

```
[edit interfaces]
```

```
user@leaf-2# set ae0 esi 00:01:01:01:01:01:01:01:01:01
user@leaf-2# set ae0 esi all-active
user@leaf-2# set ae0 aggregated-ether-options lacp passive
user@leaf-2# set ae0 aggregated-ether-options lacp system-id 00:00:00:01:01:01
user@leaf-2# set ae0 unit 0 family ethernet-switching interface-mode access
user@leaf-2# set ae0 unit 0 family ethernet-switching vlan members v100
user@leaf-2# set ae1 esi 00:02:02:02:02:02:02:02:02:02
user@leaf-2# set ae1 esi all-active
user@leaf-2# set ae1 aggregated-ether-options lacp passive
user@leaf-2# set ae1 aggregated-ether-options lacp system-id 00:00:00:01:01:01
user@leaf-2# set ae1 unit 0 family ethernet-switching interface-mode access
user@leaf-2# set ae1 unit 0 family ethernet-switching vlan members v200
```

14. Configure the loopback interface address.

```
[edit interfaces]
```

```
user@leaf-2# set lo0 unit 0 family inet address 10.255.255.5/32
```

Configuring Spine-1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

The configuration of the spine switches is similar to the leaf switches. To configure Spine-1:

1. Set the system hostname.

```
[edit]
```

```
user@spine-1# set system host-name spine-1
```

2. Configure the routing options.

```
[edit routing-options]
```

```
user@spine-1# set router-id 10.255.255.2
user@spine-1# set autonomous-system 65401
user@spine-1# set forwarding-table export load-balance
user@spine-1# set forwarding-table ecmp-fast-reroute
```

3. Configure load balance policy options.

```
[edit policy-options policy-statement load-balance]
```

```
user@spine-1# set term 1 then load-balance per-packet
```

4. Configure the underlay Leaf group with the **advertise-peer-as** statement to enable Spine-1 to bypass EBGP rules and re-advertise a route to the same autonomous system (AS) number.

```
[edit protocols bgp group underlay-leaf]
```

```
user@spine-1# set type external
user@spine-1# set advertise-peer-as
user@spine-1# set family inet unicast loops 2
user@spine-1# set export lo0
user@spine-1# set peer-as 65402
user@spine-1# set multipath
user@spine-1# set neighbor 10.0.0.9 description leaf-1
user@spine-1# set neighbor 10.0.0.11 description leaf-2
```

5. Configure the underlay Core group. The MX Series Core routers are not EVPN peered, and do not require reachability to the other's loopback, so including the **advertise-peer-as** configuration statement is optional.

```
[edit protocols bgp group underlay-core]
```

```
user@spine-1# set type external
user@spine-1# set advertise-peer-as
user@spine-1# set family inet unicast loops 2
user@spine-1# set export lo0
user@spine-1# set peer-as 65400
user@spine-1# set multipath
user@spine-1# set neighbor 10.0.0.0 description core-1
user@spine-1# set neighbor 10.0.0.4 description core-2
```

6. Configure policy options for the loopback address.

```
[edit policy-options policy-statement lo0]
```

```
user@spine-1# set from family inet
user@spine-1# set from protocol direct
user@spine-1# set from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@spine-1# set then accept
```

Configuring Spine-2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

The configuration of the spine switches is similar to the leaf switches. To configure Spine-2:

1. Set the system hostname.

```
[edit]
```

```
user@spine-2# set system host-name spine-2
```

2. Configure the routing options.

```
[edit routing-options]
```

```
user@spine-2# set router-id 10.255.255.3
user@spine-2# set autonomous-system 65401
user@spine-2# set forwarding-table export load-balance
user@spine-2# set forwarding-table ecmp-fast-reroute
```

3. Configure load balance policy options.

```
[edit policy-options policy-statement load-balance]
```

```
user@spine-2# set term 1 then load-balance per-packet
```

4. Configure the underlay Leaf group with the **advertise-peer-as** statement to enable Spine-2 to bypass EBGP rules and re-advertise a route to the same autonomous system (AS) number.

```
[edit protocols bgp group underlay-leaf]
```

```
user@spine-2# set type external
user@spine-2# set advertise-peer-as
user@spine-2# set family inet unicast loops 2
user@spine-2# set export lo0
user@spine-2# set peer-as 65402
user@spine-2# set multipath
```

```
user@spine-2# set neighbor 10.0.0.13 description leaf-1
user@spine-2# set neighbor 10.0.0.15 description leaf-2
```

5. Configure the underlay Core group. The MX Series Core routers are not EVPN peered, and do not require reachability to the other's loopback, so including the **advertise-peer-as** configuration statement is optional.

```
[edit protocols bgp group underlay-core]
```

```
user@spine-2# set type external
user@spine-2# set advertise-peer-as
user@spine-2# set family inet unicast loops 2
user@spine-2# set export lo0
user@spine-2# set peer-as 65400
user@spine-2# set multipath
user@spine-2# set neighbor 10.0.0.2 description core-1
user@spine-2# set neighbor 10.0.0.6 description core-2
```

6. Configure policy options for the loopback address.

```
[edit policy-options policy-statement lo0]
```

```
user@spine-2# set from family inet
user@spine-2# set from protocol direct
user@spine-2# set from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@spine-2# set then accept
```

Configuring Core-1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Core-1:

1. Set the system hostname.

```
[edit]
```

```
user@core-1# set system host-name core-1
```

2. Configure the router ID and other routing options.

```
[edit routing-options]
```

```
user@core-1# set router-id 10.255.255.2
user@core-1# set autonomous-system 65400
user@core-1# set forwarding-table export load-balance
user@core-1# set forwarding-table ecmp-fast-reroute
```

3. Configure the load balance policy options.

```
[edit policy-options policy-statement load-balance]
```

```
user@core-1# set term 1 then load-balance per-packet
```

4. Configure the BGP underlay group.

```
[edit protocols bgp group underlay]
```

```
user@core-1# set type external
user@core-1# set advertise-peer-as
user@core-1# set family inet unicast loops 2
user@core-1# set export lo0
user@core-1# set peer-as 65401
user@core-1# set multipath
user@core-1# set neighbor 10.0.0.1 description spine-1
user@core-1# set neighbor 10.0.0.3 description spine-2
```

5. Configure the loopback address policy options.

```
[edit policy-options policy-statement lo0]
```

```
user@core-1# set from family inet
user@core-1# set from protocol direct
user@core-1# set lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@core-1# set lo0 then accept
```

6. A large portion of Core-1's configuration takes place in the **[routing-instance]** hierarchy. Configure the virtual routers and configure a unique VRF table import policy for each virtual switch.

```
[edit routing-instances]
```

```
user@core-1# set VRF_Tenant_A instance-type vrf
user@core-1# set VRF_Tenant_A interface irb.1100
user@core-1# set VRF_Tenant_A route-distinguisher 10.255.255.2:1100
user@core-1# set VRF_Tenant_A vrf-target target:10:100
user@core-1# set VRF_Tenant_A routing-options auto-export
user@core-1# set VRF_Tenant_B instance-type vrf
user@core-1# set VRF_Tenant_B interface irb.1200
user@core-1# set VRF_Tenant_B route-distinguisher 10.255.255.2:1200
user@core-1# set VRF_Tenant_B vrf-target target:10:200
user@core-1# set VRF_Tenant_C instance-type vrf
user@core-1# set VRF_Tenant_C interface irb.1300
user@core-1# set VRF_Tenant_C route-distinguisher 10.255.255.2:1300
user@core-1# set VRF_Tenant_C vrf-target target:10:300
user@core-1# set VRF_Tenant_D instance-type vrf
user@core-1# set VRF_Tenant_D interface irb.1400
user@core-1# set VRF_Tenant_D route-distinguisher 10.255.255.2:1400
user@core-1# set VRF_Tenant_D vrf-target target:10:400
user@core-1# set VS_VLAN100 vtep-source-interface lo0.0
user@core-1# set VS_VLAN100 instance-type virtual-switch
user@core-1# set VS_VLAN100 route-distinguisher 10.255.255.2:100
user@core-1# set VS_VLAN100 vrf-import VS_VLAN100_IMP
user@core-1# set VS_VLAN100 vrf-target target:1:100
user@core-1# set VS_VLAN100 protocols evpn encapsulation vxlan
user@core-1# set VS_VLAN100 protocols evpn extended-vni-list 1100
user@core-1# set VS_VLAN100 protocols evpn multicast-mode ingress-replication
user@core-1# set VS_VLAN100 bridge-domains bd1100 vlan-id 100
user@core-1# set VS_VLAN100 bridge-domains bd1100 routing-interface irb.1100
user@core-1# set VS_VLAN100 bridge-domains bd1100 vxlan vni 1100
user@core-1# set VS_VLAN100 bridge-domains bd1100 vxlan
ingress-node-replication
user@core-1# set VS_VLAN200 vtep-source-interface lo0.0
user@core-1# set VS_VLAN200 instance-type virtual-switch
user@core-1# set VS_VLAN200 route-distinguisher 10.255.255.2:200
user@core-1# set VS_VLAN200 vrf-import VS_VLAN200_IMP
user@core-1# set VS_VLAN200 vrf-target target:1:200
user@core-1# set VS_VLAN200 protocols evpn encapsulation vxlan
user@core-1# set VS_VLAN200 protocols evpn extended-vni-list 1200
user@core-1# set VS_VLAN200 protocols evpn multicast-mode ingress-replication
user@core-1# set VS_VLAN200 bridge-domains bd1200 vlan-id 200
user@core-1# set VS_VLAN200 bridge-domains bd1200 routing-interface irb.1200
user@core-1# set VS_VLAN200 bridge-domains bd1200 vxlan vni 1200
user@core-1# set VS_VLAN200 bridge-domains bd1200 vxlan
ingress-node-replication
user@core-1# set VS_VLAN300 vtep-source-interface lo0.0
user@core-1# set VS_VLAN300 instance-type virtual-switch
user@core-1# set VS_VLAN300 route-distinguisher 10.255.255.2:300
user@core-1# set VS_VLAN300 vrf-import VS_VLAN300_IMP
user@core-1# set VS_VLAN300 vrf-target target:1:300
user@core-1# set VS_VLAN300 protocols evpn encapsulation vxlan
user@core-1# set VS_VLAN300 protocols evpn extended-vni-list 1300
user@core-1# set VS_VLAN300 protocols evpn multicast-mode ingress-replication
user@core-1# set VS_VLAN300 bridge-domains bd1300 vlan-id 300
user@core-1# set VS_VLAN300 bridge-domains bd1300 routing-interface irb.1300
user@core-1# set VS_VLAN300 bridge-domains bd1300 vxlan vni 1300
user@core-1# set VS_VLAN300 bridge-domains bd1300 vxlan
ingress-node-replication
user@core-1# set VS_VLAN400 vtep-source-interface lo0.0
user@core-1# set VS_VLAN400 instance-type virtual-switch
user@core-1# set VS_VLAN400 route-distinguisher 10.255.255.2:400
```

```
user@core-1# set VS_VLAN400 vrf-import VS_VLAN400_IMP
user@core-1# set VS_VLAN400 vrf-target target:1:400
user@core-1# set VS_VLAN400 protocols evpn encapsulation vxlan
user@core-1# set VS_VLAN400 protocols evpn extended-vni-list 1400
user@core-1# set VS_VLAN400 protocols evpn multicast-mode ingress-replication
user@core-1# set VS_VLAN400 bridge-domains bd1400 vlan-id 400
user@core-1# set VS_VLAN400 bridge-domains bd1400 routing-interface irb.1400
user@core-1# set VS_VLAN400 bridge-domains bd1400 vxlan vni 1400
user@core-1# set VS_VLAN400 bridge-domains bd1400 vxlan
ingress-node-replication
```

7. Configure the individual policy statements.

```
[edit policy-options policy-statement]
```

```
user@core-1# set VS_VLAN100_IMP term ESI from community comm-leaf_esi
user@core-1# set VS_VLAN100_IMP term ESI then accept
user@core-1# set VS_VLAN100_IMP term VS_VLAN100 from community comm-VS_VLAN100
user@core-1# set VS_VLAN100_IMP term VS_VLAN100 then accept
user@core-1# set VS_VLAN200_IMP term ESI from community comm-leaf_esi
user@core-1# set VS_VLAN200_IMP term ESI then accept
user@core-1# set VS_VLAN200_IMP term VS_VLAN200 from community comm-VS_VLAN200
user@core-1# set VS_VLAN200_IMP term VS_VLAN200 then accept
user@core-1# set VS_VLAN300_IMP term ESI from community comm-leaf_esi
user@core-1# set VS_VLAN300_IMP term ESI then accept
user@core-1# set VS_VLAN300_IMP term VS_VLAN300 from community comm-VS_VLAN300
user@core-1# set VS_VLAN300_IMP term VS_VLAN300 then accept
user@core-1# set VS_VLAN400_IMP term ESI from community comm-leaf_esi
user@core-1# set VS_VLAN400_IMP term ESI then accept
user@core-1# set VS_VLAN400_IMP term VS_VLAN400 from community comm-VS_VLAN400
user@core-1# set VS_VLAN400_IMP term VS_VLAN400 then accept
user@core-1# set lo0 from family inet
user@core-1# set lo0 from protocol direct
user@core-1# set lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@core-1# set lo0 then accept
```

8. Configure the community policy options. Make sure the **comm-leaf_esi** policy option accepts target 9999:9999 to ensure that all virtual switches import the Type-1 ESI routes from all leafs.

```
[edit policy-options community]
```

```
user@core-1# set comm-VS_VLAN100 members target:1:100
user@core-1# set comm-VS_VLAN200 members target:1:200
user@core-1# set comm-VS_VLAN300 members target:1:300
user@core-1# set comm-VS_VLAN400 members target:1:400
user@core-1# set comm-leaf_esi members target:9999:9999
```

9. Configure the IRB interfaces. Every IRB has a virtual gateway address, which is a shared MAC address and IP address across Core-1 and Core-2.

```
[edit interfaces irb]
```

```
user@core-1# set unit 1100 family inet address 172.16.0.2/24
virtual-gateway-address 100.0.0.1
user@core-1# set unit 1200 family inet address 172.16.0.3/24
virtual-gateway-address 200.0.0.1
user@core-1# set unit 1300 family inet address 10.10.10.2/24
virtual-gateway-address 10.10.10.1
user@core-1# set unit 1400 family inet address 10.10.10.2/24
virtual-gateway-address 10.10.10.1
```

10. Configure MP-BGP sessions towards Leaf-1 and Leaf-2.

```
[edit protocols bgp group EVPN_VXLAN]
```

```
user@core-1# set type external
user@core-1# set local-address 10.255.255.2
user@core-1# set family evpn signaling
user@core-1# set peer-as 65402
user@core-1# set multipath
user@core-1# set neighbor 192.0.2.2 description leaf-1
user@core-1# set neighbor 192.0.2.2 multihop ttl 255
user@core-1# set neighbor 10.255.255.5 description leaf-2
user@core-1# set neighbor 10.255.255.5 multihop ttl 255
```

Configuring Core-2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Core-2:

1. Set the system hostname.

```
[edit]
```

```
user@core-2# set system host-name core-2
```

2. Configure the router ID and other routing options.

```
[edit routing-options]
```

```
user@core-2# set router-id 10.255.255.1
user@core-2# set autonomous-system 65400
user@core-2# set forwarding-table export load-balance
user@core-2# set forwarding-table ecmp-fast-reroute
```

3. Configure the load balance policy options.

```
[edit policy-options policy-statement load-balance]
```

```
user@core-2# set term 1 then load-balance per-packet
```

4. Configure the BGP underlay group.

```
[edit protocols bgp group underlay]
```

```
user@core-2# set type external
user@core-2# set advertise-peer-as
user@core-2# set family inet unicast loops 2
user@core-2# set export lo0
user@core-2# set peer-as 65401
user@core-2# set multipath
user@core-2# set neighbor 10.0.0.5 description spine-1
user@core-2# set neighbor 10.0.0.7 description spine-2
```

5. Configure the loopback address policy options.

```
[edit policy-options policy-statement lo0]
```

```
user@core-2# set from family inet
user@core-2# set from protocol direct
user@core-2# set lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@core-2# set lo0 then accept
```

6. A large portion of Core-1's configuration takes place in the **[routing-instance]** hierarchy. Configure the virtual routers and configure a unique VRF table import policy for each virtual switch.

```
[edit routing-instances]
```

```
user@core-2# set VRF_Tenant_A instance-type vrf
user@core-2# set VRF_Tenant_A interface irb.1100
user@core-2# set VRF_Tenant_A route-distinguisher 10.255.255.1:1100
```

```
user@core-2# set VRF_Tenant_A vrf-target target:10:100
user@core-2# set VRF_Tenant_A routing-options auto-export
user@core-2# set VRF_Tenant_B instance-type vrf
user@core-2# set VRF_Tenant_B interface irb.1200
user@core-2# set VRF_Tenant_B route-distinguisher 10.255.255.1:1200
user@core-2# set VRF_Tenant_B vrf-target target:10:200
user@core-2# set VRF_Tenant_C instance-type vrf
user@core-2# set VRF_Tenant_C interface irb.1300
user@core-2# set VRF_Tenant_C route-distinguisher 10.255.255.1:1300
user@core-2# set VRF_Tenant_C vrf-target target:10:300
user@core-2# set VRF_Tenant_D instance-type vrf
user@core-2# set VRF_Tenant_D interface irb.1400
user@core-2# set VRF_Tenant_D route-distinguisher 10.255.255.1:1400
user@core-2# set VRF_Tenant_D vrf-target target:10:400
user@core-2# set VS_VLAN100 vtep-source-interface lo0.0
user@core-2# set VS_VLAN100 instance-type virtual-switch
user@core-2# set VS_VLAN100 route-distinguisher 10.255.255.1:100
user@core-2# set VS_VLAN100 vrf-import VS_VLAN100_IMP
user@core-2# set VS_VLAN100 vrf-target target:1:100
user@core-2# set VS_VLAN100 protocols evpn encapsulation vxlan
user@core-2# set VS_VLAN100 protocols evpn extended-vni-list 1100
user@core-2# set VS_VLAN100 protocols evpn multicast-mode ingress-replication
user@core-2# set VS_VLAN100 protocols evpn default-gateway
no-gateway-community
user@core-2# set VS_VLAN100 bridge-domains bd1100 vlan-id 100
user@core-2# set VS_VLAN100 bridge-domains bd1100 routing-interface irb.1100
user@core-2# set VS_VLAN100 bridge-domains bd1100 vxlan vni 1100
user@core-2# set VS_VLAN100 bridge-domains bd1100 vxlan
ingress-node-replication
user@core-2# set VS_VLAN200 vtep-source-interface lo0.0
user@core-2# set VS_VLAN200 instance-type virtual-switch
user@core-2# set VS_VLAN200 route-distinguisher 10.255.255.1:200
user@core-2# set VS_VLAN200 vrf-import VS_VLAN200_IMP
user@core-2# set VS_VLAN200 vrf-target target:1:200
user@core-2# set VS_VLAN200 protocols evpn encapsulation vxlan
user@core-2# set VS_VLAN200 protocols evpn extended-vni-list 1200
user@core-2# set VS_VLAN200 protocols evpn multicast-mode ingress-replication
user@core-2# set VS_VLAN200 bridge-domains bd1200 vlan-id 200
user@core-2# set VS_VLAN200 bridge-domains bd1200 routing-interface irb.1200
user@core-2# set VS_VLAN200 bridge-domains bd1200 vxlan vni 1200
user@core-2# set VS_VLAN200 bridge-domains bd1200 vxlan
ingress-node-replication
user@core-2# set VS_VLAN300 vtep-source-interface lo0.0
user@core-2# set VS_VLAN300 instance-type virtual-switch
user@core-2# set VS_VLAN300 route-distinguisher 10.255.255.1:300
user@core-2# set VS_VLAN300 vrf-import VS_VLAN300_IMP
user@core-2# set VS_VLAN300 vrf-target target:1:300
user@core-2# set VS_VLAN300 protocols evpn encapsulation vxlan
user@core-2# set VS_VLAN300 protocols evpn extended-vni-list 1300
user@core-2# set VS_VLAN300 protocols evpn multicast-mode ingress-replication
user@core-2# set VS_VLAN300 bridge-domains bd1300 vlan-id 300
user@core-2# set VS_VLAN300 bridge-domains bd1300 routing-interface irb.1300
user@core-2# set VS_VLAN300 bridge-domains bd1300 vxlan vni 1300
user@core-2# set VS_VLAN300 bridge-domains bd1300 vxlan
ingress-node-replication
user@core-2# set VS_VLAN400 vtep-source-interface lo0.0
user@core-2# set VS_VLAN400 instance-type virtual-switch
user@core-2# set VS_VLAN400 route-distinguisher 10.255.255.1:400
user@core-2# set VS_VLAN400 vrf-import VS_VLAN400_IMP
```

```
user@core-2# set VS_VLAN400 vrf-target target:1:400
user@core-2# set VS_VLAN400 protocols evpn encapsulation vxlan
user@core-2# set VS_VLAN400 protocols evpn extended-vni-list 1400
user@core-2# set VS_VLAN400 protocols evpn multicast-mode ingress-replication
user@core-2# set VS_VLAN400 bridge-domains bd1400 vlan-id 400
user@core-2# set VS_VLAN400 bridge-domains bd1400 routing-interface irb.1400
user@core-2# set VS_VLAN400 bridge-domains bd1400 vxlan vni 1400
user@core-2# set VS_VLAN400 bridge-domains bd1400 vxlan
ingress-node-replication
```

7. Configure the individual policy statements.

```
[edit policy-options policy-statement]
```

```
user@core-2# set VS_VLAN100_IMP term ESI from community comm-leaf_esi
user@core-2# set VS_VLAN100_IMP term ESI then accept
user@core-2# set VS_VLAN100_IMP term VS_VLAN100 from community comm-VS_VLAN100
user@core-2# set VS_VLAN100_IMP term VS_VLAN100 then accept
user@core-2# set VS_VLAN200_IMP term ESI from community comm-leaf_esi
user@core-2# set VS_VLAN200_IMP term ESI then accept
user@core-2# set VS_VLAN200_IMP term VS_VLAN200 from community comm-VS_VLAN200
user@core-2# set VS_VLAN200_IMP term VS_VLAN200 then accept
user@core-2# set VS_VLAN300_IMP term ESI from community comm-leaf_esi
user@core-2# set VS_VLAN300_IMP term ESI then accept
user@core-2# set VS_VLAN300_IMP term VS_VLAN300 from community comm-VS_VLAN300
user@core-2# set VS_VLAN300_IMP term VS_VLAN300 then accept
user@core-2# set VS_VLAN400_IMP term ESI from community comm-leaf_esi
user@core-2# set VS_VLAN400_IMP term ESI then accept
user@core-2# set VS_VLAN400_IMP term VS_VLAN400 from community comm-VS_VLAN400
user@core-2# set VS_VLAN400_IMP term VS_VLAN400 then accept
user@core-2# set lo0 from family inet
user@core-2# set lo0 from protocol direct
user@core-2# set lo0 from route-filter 0.0.0.0/0 prefix-length-range /32-/32
user@core-2# set lo0 then accept
```

8. Configure the community policy options. Make sure the **comm-leaf_esi** policy option accepts target 9999:9999 to ensure that all virtual switches import the Type-1 ESI routes from all leafs.

```
[edit policy-options community]
```

```
user@core-2# set comm-VS_VLAN100 members target:1:100
user@core-2# set comm-VS_VLAN200 members target:1:200
user@core-2# set comm-VS_VLAN300 members target:1:300
user@core-2# set comm-VS_VLAN400 members target:1:400
user@core-2# set comm-leaf_esi members target:9999:9999
```

9. Configure the IRB interfaces. Every IRB has a virtual gateway address, which is a shared MAC address and IP address across Core-1 and Core-2.

```
[edit interfaces irb]
```

```
user@core-2# set unit 1100 family inet address 172.16.0.4/24
virtual-gateway-address 100.0.0.1
user@core-2# set unit 1200 family inet address 172.16.0.5/24
virtual-gateway-address 200.0.0.1
user@core-2# set unit 1300 family inet address 10.10.10.3/24
virtual-gateway-address 10.10.10.1
user@core-2# set unit 1400 family inet address 10.10.10.3/24
virtual-gateway-address 10.10.10.1
```

10. Configure MP-BGP sessions towards Leaf-1 and Leaf-2.

```
[edit protocols bgp group EVPN_VXLAN]
```

```
user@core-2# set type external
user@core-2# set local-address 10.255.255.1
user@core-2# set family evpn signaling
user@core-2# set peer-as 65402
user@core-2# set multipath
user@core-2# set neighbor 192.0.2.2 description leaf-1
user@core-2# set neighbor 192.0.2.2 multihop ttl 255
user@core-2# set neighbor 10.255.255.5 description leaf-2
user@core-2# set neighbor 10.255.255.5 multihop ttl 255
```

Verification

After you configure both the underlay and EVPN overlay we recommend that you verify that the configurations work as you intended.

- Verifying Leaf-1 Configuration on page 405
- Verifying Leaf-2 Configuration on page 409
- Verifying Spine-1 Configuration on page 414
- Verifying Spine-2 Configuration on page 417
- Verifying Core-1 Configuration on page 419
- Verifying Core-2 Configuration on page 426
- Verifying MAC Reachability to a Single-Homed CE Device (Leaf-1) on page 433
- Verifying MAC Reachability to a Single-Homed CE Device (Type-2) on page 434
- Verifying the Imported Route on page 435
- Verifying the Layer 2 Address Learning Daemon Copy on page 437
- Verifying the Kernel-Level Forwarding Table on page 438
- Verifying MAC Reachability to a Multihomed CE Device on page 439

- Verifying EVPN, Layer 2 Address Learning Daemon, and the Kernel-Forwarding Tables on page 441
- Verifying an EVPN Anycast Gateway on page 446

Verifying Leaf-1 Configuration

Purpose Verify that Leaf-1 is properly configured.

Action Verify that the routing options are properly configured.

```
user@leaf-1> show configuration routing-options
```

```
router-id 192.0.2.2;
autonomous-system 65402;
forwarding-table {
    export load-balance;
}
```

Verify that the load-balance policy statement is properly configured.

```
user@leaf-1> show configuration policy-options policy-statement
```

```
load-balance
term 1 {
    then {
        load-balance per-packet;
    }
}
```

Verify the underlay BGP group configuration.

```
user@leaf-1> show configuration protocols bgp group underlay
```

```
type external;
advertise-peer-as;
family inet {
    unicast {
        loops 2;
    }
}
export lo0;
peer-as 65401;
multipath;
neighbor 10.0.0.8 {
    description spine-1;
}
neighbor 10.0.0.12 {
    description spine-2;
}
```

Verify that the loopback address is properly configured.

```
user@leaf-1> show configuration policy-options policy-statement lo0
```

```
from {
    family inet;
    protocol direct;
    route-filter 0.0.0.0/0 prefix-length-range /32-/32;
}
then accept;
```

Verify the configuration of the switch options.

```
user@leaf-1> show configuration switch-options
```

```
vtep-source-interface lo0.0;
route-distinguisher 192.0.2.2:1;
vrf-import vrf-imp;
vrf-target target:9999:9999;
```

Verify the configuration of the VRF table import policy statement.

```
user@leaf-1> show configuration policy-options policy-statement
```

```
vrf-imp
term t1 {
    from community com100;
    then accept;
}
term t2 {
    from community com200;
    then accept;
}
term t3 {
    from community com300;
    then accept;
}
term t4 {
    from community com400;
    then accept;
}
term t5 {
    then reject;
}
```

Verify the policy options for GREP.

```
user@leaf-1> show configuration policy-options | grep members
```

```
community com100 members target:1:100;
community com200 members target:1:200;
community com300 members target:1:300;
community com400 members target:1:400;
```

Verify that EVPN is properly configured with VXLAN details.

```
user@leaf-1> show configuration protocols evpn
```

```
encapsulation vxlan;
extended-vni-list [ 1100 1200 1300 1400 ];
multicast-mode ingress-replication;
vni-routing-options {
    vni 1100 {
        vrf-target export target:1:100;
    }
    vni 1200 {
```

```
        vrf-target export target:1:200;
    }
    vni 1300 {
        vrf-target export target:1:300;
    }
    vni 1400 {
        vrf-target export target:1:400;
    }
}
```

Verify the EVPN-VXLAN Core BGP group configuration.

```
user@leaf-1> show configuration protocols bgp group EVPN_VXLAN_CORE
```

```
type external;
multihop {
    ttl 255;
    no-nexthop-change;
}
local-address 192.0.2.2;
family evpn {
    signaling;
}
peer-as 65400;
neighbor 10.255.255.2 {
    description core-1;
}
neighbor 10.255.255.1 {
    description core-2;
}
```

Verify the EVPN-VXLAN Leaf BGP group configuration.

```
user@leaf-1> show configuration protocols bgp group EVPN_VXLAN_LEAF
```

```
type internal;
local-address 192.0.2.2;
family evpn {
    signaling;
}
neighbor 10.255.255.5 {
    description leaf-2;
}
```

Verify the configuration for the xe-0/0/32 interface.

```
user@leaf-1> show configuration interfaces xe-0/0/32
```

```
ether-options {
    802.3ad ae0;
}
```

Verify the configuration for the xe-0/0/33 interface.

```
user@leaf-1> show configuration interfaces xe-0/0/33
```

```
ether-options {
    802.3ad ae1;
}
```

Verify the configuration for the ae0 interface.

```
user@leaf-1> show configuration interfaces ae0
```

```
esi {
    00:01:01:01:01:01:01:01:01:01;
    all-active;
}
aggregated-ether-options {
    lacp {
        active;
        system-id 00:00:00:01:01:01;
    }
}
unit 0 {
    family ethernet-switching {
        interface-mode access;
        vlan {
            members v100;
        }
    }
}
```

Verify the configuration for the ae1 interface.

```
user@leaf-1> show configuration interfaces ae1
```

```
esi {
    00:02:02:02:02:02:02:02:02:02;
    all-active;
}
aggregated-ether-options {
    lacp {
        active;
        system-id 00:00:00:01:01:01;
    }
}
unit 0 {
    family ethernet-switching {
        interface-mode access;
        vlan {
            members v200;
        }
    }
}
```

Verifying Leaf-2 Configuration

Purpose Verify that Leaf-2 is properly configured.

Action Verify that the routing options are properly configured.

```
user@leaf-2> show configuration routing-options
```

```
router-id 10.255.255.5;
autonomous-system 65402;
forwarding-table {
    export load-balance;
}
```

Verify that the load-balance policy statement is properly configured.

```
user@leaf-2> show configuration policy-options policy-statement
```

```
load-balance
term 1 {
    then {
        load-balance per-packet;
    }
}
```

Verify the underlay BGP group configuration.

```
user@leaf-2> show configuration protocols bgp group underlay
```

```
type external;
advertise-peer-as;
family inet {
    unicast {
        loops 2;
    }
}
export lo0;
peer-as 65401;
multipath;
neighbor 10.0.0.10 {
    description spine-1;
}
neighbor 10.0.0.14 {
    description spine-2;
}
```

Verify that the loopback address is properly configured.

```
user@leaf-2> show configuration policy-options policy-statement lo0
```

```
from {
    family inet;
    protocol direct;
    route-filter 0.0.0.0/0 prefix-length-range /32-/32;
}
then accept;
```

Verify the configuration of the switch options.

```
user@leaf-2> show configuration switch-options
```

```
vtep-source-interface lo0.0;
route-distinguisher 10.255.255.5:1;
vrf-import vrf-imp;
vrf-target target:9999:9999;
```

Verify the configuration of the VRF table import policy statement.

```
user@leaf-2> show configuration policy-options policy-statement
```

```
vrf-imp
term t1 {
    from community com100;
    then accept;
}
term t2 {
    from community com200;
    then accept;
}
term t3 {
    from community com300;
    then accept;
}
term t4 {
    from community com400;
    then accept;
}
term t5 {
    then reject;
}
```

Verify the policy options for GREP.

```
user@leaf-2> show configuration policy-options | grep members
```

```
community com100 members target:1:100;
community com200 members target:1:200;
community com300 members target:1:300;
community com400 members target:1:400;
```

Verify that EVPN is properly configured with VXLAN details.

```
user@leaf-2> show configuration protocols evpn
```

```
encapsulation vxlan;
extended-vni-list [ 1100 1200 1300 1400 ];
multicast-mode ingress-replication;
vni-routing-options {
    vni 1100 {
        vrf-target export target:1:100;
    }
    vni 1200 {
```

```
        vrf-target export target:1:200;
    }
    vni 1300 {
        vrf-target export target:1:300;
    }
    vni 1400 {
        vrf-target export target:1:400;
    }
}
```

Verify the EVPN-VXLAN Core BGP group configuration.

```
user@leaf-2> show configuration protocols bgp group EVPN_VXLAN_CORE
```

```
type external;
multihop {
    ttl 255;
    no-nexthop-change;
}
local-address 10.255.255.5;
family evpn {
    signaling;
}
peer-as 65400;
neighbor 10.255.255.2 {
    description core-1;
}
neighbor 10.255.255.1 {
    description core-2;
}
```

Verify the EVPN-VXLAN Leaf BGP group configuration.

```
user@leaf-2> show configuration protocols bgp group EVPN_VXLAN_LEAF
```

```
type internal;
local-address 192.0.2.2;
family evpn {
    signaling;
}
neighbor 10.255.255.5 {
    description leaf-2;
}
```

Verify the configuration for the xe-0/0/36 interface.

```
user@leaf-2> show configuration interfaces xe-0/0/36
```

```
ether-options {
    802.3ad ae0;
}
```

Verify the configuration for the xe-0/0/37 interface.

```
user@leaf-2> show configuration interfaces xe-0/0/37
```

```
ether-options {
    802.3ad ae1;
}
```

Verify the configuration for the ae0 interface.

```
user@leaf-2> show configuration interfaces ae0
```

```
esi {
    00:01:01:01:01:01:01:01:01:01;
    all-active;
}
aggregated-ether-options {
    lacp {
        passive;
        system-id 00:00:00:01:01:01;
    }
}
unit 0 {
    family ethernet-switching {
        interface-mode access;
        vlan {
            members v100;
        }
    }
}
```

Verify the configuration for the ae1 interface.

```
user@leaf-2> show configuration interfaces ae1
```

```
esi {
    00:02:02:02:02:02:02:02:02:02;
    all-active;
}
aggregated-ether-options {
    lacp {
        passive;
        system-id 00:00:00:01:01:01;
    }
}
unit 0 {
    family ethernet-switching {
        interface-mode access;
        vlan {
            members v200;
        }
    }
}
```

Verifying Spine-1 Configuration

Purpose Verify that Spine-1 is properly configured.

Action Verify that the routing options are properly configured.

```
user@spine-1> show configuration routing-options
```

```
router-id 10.255.255.2;
autonomous-system 65401;
forwarding-table {
    export load-balance;
    ecmp-fast-reroute;
}
```

Verify that the load-balance policy statement is properly configured.

```
user@spine-1> show configuration policy-options policy-statement load-balance
```

```
term 1 {
    then {
        load-balance per-packet;
    }
}
```

Verify the underlay Leaf BGP group configuration.

```
user@spine-1> show configuration protocols bgp group underlay-leaf
```

```
type external;
advertise-peer-as;
family inet {
    unicast {
        loops 2;
    }
}
export lo0;
peer-as 65402;
multipath;
neighbor 10.0.0.9 {
    description leaf-1;
}
neighbor 10.0.0.11 {
    description leaf-2;
}
```

Verify the underlay Core BGP group configuration.

```
user@spine-1> show configuration protocols bgp group underlay-core
```

```
type external;
advertise-peer-as;
family inet {
    unicast {
```

```
        loops 2;
    }
}
export lo0;
peer-as 65400;
multipath;
neighbor 10.0.0.0 {
    description core-1;
}
neighbor 10.0.0.4 {
    description core-2;
}
```

Verify that the loopback address is properly configured.

```
user@spine-1> show configuration policy-options policy-statement lo0
```

```


from {
    family inet;
    protocol direct;
    route-filter 0.0.0.0/0 prefix-length-range /32-/32;
}
then accept;
```

Verifying Spine-2 Configuration

Purpose Verify that Spine-2 is properly configured.

Action Verify that the routing options are properly configured.

```
user@spine-2> show configuration routing-options
```

```
router-id 10.255.255.3;
autonomous-system 65401;
forwarding-table {
    export load-balance;
    ecmp-fast-reroute;
}
```

Verify that the load-balance policy statement is properly configured.

```
user@spine-2> show configuration policy-options policy-statement load-balance
```

```
term 1 {
    then {
        load-balance per-packet;
    }
}
```

Verify the underlay Leaf BGP group configuration.

```
user@spine-2> show configuration protocols bgp group underlay-leaf
```

```
type external;
advertise-peer-as;
family inet {
    unicast {
        loops 2;
    }
}
export lo0;
peer-as 65402;
multipath;
neighbor 10.0.0.13 {
    description leaf-1;
}
neighbor 10.0.0.15 {
    description leaf-2;
}
```

Verify the underlay Core BGP group configuration.

```
user@spine-2> show configuration protocols bgp group underlay-core
```

```
type external;
advertise-peer-as;
family inet {
    unicast {
```

```
        loops 2;
    }
}
export lo0;
peer-as 65400;
multipath;
neighbor 10.0.0.2 {
    description core-1;
}
neighbor 10.0.0.6 {
    description core-2;
}
```

Verify that the loopback address is properly configured.

```
user@spine-2> show configuration policy-options policy-statement lo0
```

```

from {
    family inet;
    protocol direct;
    route-filter 0.0.0.0/0 prefix-length-range /32-/32;
}
then accept;
```

Verifying Core-1 Configuration

Purpose Verify that Core-1 is properly configured.

Action Verify that the routing options are properly configured.

```
user@core-1> show configuration routing-options
```

```
router-id 10.255.255.2;
autonomous-system 65400;
forwarding-table {
    export load-balance;
    ecmp-fast-reroute;
}
```

Verify that the load-balance policy statement is properly configured.

```
user@core-1> show configuration policy-options policy-statement load-balance
```

```
term 1 {
    then {
        load-balance per-packet;
    }
```

Verify the underlay BGP group configuration.

```
user@core-1> show configuration protocols bgp
```

```
group underlay {
    type external;
    advertise-peer-as;
    family inet {
        unicast {
            loops 2;
        }
    }
    export lo0;
    peer-as 65401;
    multipath;
    neighbor 10.0.0.1 {
        description spine-1;
    }
    neighbor 10.0.0.3 {
        description spine-2;
    }
}
```

Verify that the loopback address is properly configured.

```
user@core-1> show configuration policy-options policy-statement lo0
```

```
from {
    family inet;
    protocol direct;
    route-filter 0.0.0.0/0 prefix-length-range /32-/32;
}
then accept;
```

Verify that the routing instances are properly configured.

```
user@core-1> show configuration routing-instances

VRF_Tenant_A {
    instance-type vrf;
    interface irb.1100;
    route-distinguisher 10.255.255.2:1100;
    vrf-target target:10:100;
}
VRF_Tenant_B {
    instance-type vrf;
    interface irb.1200;
    route-distinguisher 10.255.255.2:1200;
    vrf-target target:10:200;
}
VRF_Tenant_C {
    instance-type vrf;
    interface irb.1300;
    route-distinguisher 10.255.255.2:1300;
    vrf-target target:10:300;
}
VRF_Tenant_D {
    instance-type vrf;
    interface irb.1400;
    route-distinguisher 10.255.255.2:1400;
    vrf-target target:10:400;
}
VS_VLAN100 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.2:100;
    vrf-import VS_VLAN100_IMP;
    vrf-target target:1:100;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1100;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1100 {
            vlan-id 100;
            routing-interface irb.1100;
            vxlan {
                vni 1100;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN200 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.2:200;
    vrf-import VS_VLAN200_IMP;
    vrf-target target:1:200;
    protocols {
```

```
        evpn {
            encapsulation vxlan;
            extended-vni-list 1200;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1200 {
            vlan-id 200;
            routing-interface irb.1200;
            vxlan {
                vni 1200;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN300 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.2:300;
    vrf-import VS_VLAN300_IMP;
    bridge-domains {
        bd1100 {
            vlan-id 100;
            routing-interface irb.1100;
            vxlan {
                vni 1100;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN200 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.2:200;
    vrf-import VS_VLAN200_IMP;
    vrf-target target:1:200;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1200;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1200 {
            vlan-id 200;
            routing-interface irb.1200;
            vxlan {
                vni 1200;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN300 {
    vtep-source-interface lo0.0;
```

```
    instance-type virtual-switch;
    route-distinguisher 10.255.255.2:300;
    vrf-import VS_VLAN300_IMP;
    vrf-target target:1:300;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1300;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1300 {
            vlan-id 300;
            routing-interface irb.1300;
            vxlan {
                vni 1300;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN400 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.2:400;
    vrf-import VS_VLAN400_IMP;
    vrf-target target:1:400;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1400;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1400 {
            vlan-id 400;
            routing-interface irb.1400;
            vxlan {
                vni 1400;
                ingress-node-replication;
            }
        }


    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1300;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1300 {
            vlan-id 300;
            routing-interface irb.1300;
            vxlan {
                vni 1300;
```

```
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN400 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.2:400;
    vrf-import VS_VLAN400_IMP;
    vrf-target target:1:400;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1400;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1400 {
            vlan-id 400;
            routing-interface irb.1400;
            vxlan {
                vni 1400;
                ingress-node-replication;
            }
        }
    }
```

Verify that the policy options are properly configured.

```
user@core-1> show configuration policy-options
```

```
policy-statement VS_VLAN100_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
    }
    term VS_VLAN100 {
        from community comm-VS_VLAN100;
        then accept;
    }
}
policy-statement VS_VLAN200_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
    }
    term VS_VLAN200 {
        from community comm-VS_VLAN200;
        then accept;
    }
}
policy-statement VS_VLAN300_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
```

```
    }
    term VS_VLAN300 {
        from community comm-VS_VLAN300;
        then accept;
    }
}
policy-statement VS_VLAN400_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
    }
    term VS_VLAN400 {
        from community comm-VS_VLAN400;
        then accept;
    }
}
policy-statement lo0 {
    from {
        family inet;
        protocol direct;
        route-filter 0.0.0.0/0 prefix-length-range /32-/32;
    }
    then accept;
}
policy-statement load-balance {
    term 1 {
        then {
            load-balance per-packet;
        }
    }
}
community comm-VS_VLAN100 members target:1:100;
community comm-VS_VLAN200 members target:1:200;
community comm-VS_VLAN300 members target:1:300;
community comm-VS_VLAN400 members target:1:400;
community comm-leaf_esi members target:9999:9999;
```

Verify the configuration of the IRB interface.

```
user@core-1> show configuration interfaces irb
```

```
unit 1100 {
    family inet {
        address 172.16.0.2/24 {
            virtual-gateway-address 100.0.0.1;
        }
    }
}
unit 1200 {
    family inet {
        address 172.16.0.3/24 {
            virtual-gateway-address 200.0.0.1;
        }
    }
}
unit 1300 {
    family inet {
```

```
        address 10.10.10.2/24 {
            virtual-gateway-address 10.10.10.1;
        }
    }
}
unit 1400 {
    family inet {
        address 10.10.10.2/24 {
            virtual-gateway-address 10.10.10.1;
        }
    }
}
```

Verify the configuration of the EVPN-VXLAN BGP group.

```
user@core-1> show configuration protocols bgp group EVPN_VXLAN
```

```
type external;
local-address 10.255.255.2;
family evpn {
    signaling;
}
peer-as 65402;
multipath;
neighbor 192.0.2.2 {
    description leaf-1;
    multihop {
        ttl 255;
    }
}
neighbor 10.255.255.5 {
    description leaf-2;
    multihop {
        ttl 255;
    }
}
```

Verifying Core-2 Configuration

Purpose Verify that Core-2 is properly configured.

Action Verify that the routing options are properly configured.

```
user@core-2> show configuration routing-options
```

```
router-id 10.255.255.1;
autonomous-system 65400;
forwarding-table {
    export load-balance;
    ecmp-fast-reroute;
}
```

Verify that the load-balance policy statement is properly configured.

```
user@core-2> show configuration policy-options policy-statement load-balance
```

```
term 1 {
    then {
        load-balance per-packet;
    }
```

Verify the underlay BGP group configuration.

```
user@core-2> show configuration protocols bgp
```

```
group underlay {
    type external;
    advertise-peer-as;
    family inet {
        unicast {
            loops 2;
        }
    }
    export lo0;
    peer-as 65401;
    multipath;
    neighbor 10.0.0.5 {
        description spine-1;
    }
    neighbor 10.0.0.7 {
        description spine-2;
    }
}
```

Verify that the loopback address is properly configured.

```
user@core-2> show configuration policy-options policy-statement lo0
```

```
from {
    family inet;
    protocol direct;
    route-filter 0.0.0.0/0 prefix-length-range /32-/32;
}
then accept;
```

Verify that the routing instances are properly configured.

```
user@core-2> show configuration routing-instances

VRF_Tenant_A {
    instance-type vrf;
    interface irb.1100;
    route-distinguisher 10.255.255.1:1100;
    vrf-target target:10:100;
}
VRF_Tenant_B {
    instance-type vrf;
    interface irb.1200;
    route-distinguisher 10.255.255.1:1200;
    vrf-target target:10:200;
}
VRF_Tenant_C {
    instance-type vrf;
    interface irb.1300;
    route-distinguisher 10.255.255.1:1300;
    vrf-target target:10:300;
}
VRF_Tenant_D {
    instance-type vrf;
    interface irb.1400;
    route-distinguisher 10.255.255.1:1400;
    vrf-target target:10:400;
}
VS_VLAN100 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.1:100;
    vrf-import VS_VLAN100_IMP;
    vrf-target target:1:100;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1100;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1100 {
            vlan-id 100;
            routing-interface irb.1100;
            vxlan {
                vni 1100;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN200 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.1:200;
    vrf-import VS_VLAN200_IMP;
    vrf-target target:1:200;
    protocols {
```

```
        evpn {
            encapsulation vxlan;
            extended-vni-list 1200;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1200 {
            vlan-id 200;
            routing-interface irb.1200;
            vxlan {
                vni 1200;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN300 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.1:300;
    vrf-import VS_VLAN300_IMP;
    bridge-domains {
        bd1100 {
            vlan-id 100;
            routing-interface irb.1100;
            vxlan {
                vni 1100;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN200 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.1:200;
    vrf-import VS_VLAN200_IMP;
    vrf-target target:1:200;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1200;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1200 {
            vlan-id 200;
            routing-interface irb.1200;
            vxlan {
                vni 1200;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN300 {
    vtep-source-interface lo0.0;
```

```
    instance-type virtual-switch;
    route-distinguisher 10.255.255.1:300;
    vrf-import VS_VLAN300_IMP;
    vrf-target target:1:300;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1300;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1300 {
            vlan-id 300;
            routing-interface irb.1300;
            vxlan {
                vni 1300;
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN400 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.1:400;
    vrf-import VS_VLAN400_IMP;
    vrf-target target:1:400;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1400;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1400 {
            vlan-id 400;
            routing-interface irb.1400;
            vxlan {
                vni 1400;
                ingress-node-replication;
            }
        }


    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1300;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1300 {
            vlan-id 300;
            routing-interface irb.1300;
            vxlan {
                vni 1300;
```

```
                ingress-node-replication;
            }
        }
    }
}
VS_VLAN400 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    route-distinguisher 10.255.255.1:400;
    vrf-import VS_VLAN400_IMP;
    vrf-target target:1:400;
    protocols {
        evpn {
            encapsulation vxlan;
            extended-vni-list 1400;
            multicast-mode ingress-replication;
        }
    }
    bridge-domains {
        bd1400 {
            vlan-id 400;
            routing-interface irb.1400;
            vxlan {
                vni 1400;
                ingress-node-replication;
            }
        }
    }
```

Verify that the policy options are properly configured.

```
user@core-2> show configuration policy-options
```

```
policy-statement VS_VLAN100_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
    }
    term VS_VLAN100 {
        from community comm-VS_VLAN100;
        then accept;
    }
}
policy-statement VS_VLAN200_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
    }
    term VS_VLAN200 {
        from community comm-VS_VLAN200;
        then accept;
    }
}
policy-statement VS_VLAN300_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
```

```
    }
    term VS_VLAN300 {
        from community comm-VS_VLAN300;
        then accept;
    }
}
policy-statement VS_VLAN400_IMP {
    term ESI {
        from community comm-leaf_esi;
        then accept;
    }
    term VS_VLAN400 {
        from community comm-VS_VLAN400;
        then accept;
    }
}
policy-statement lo0 {
    from {
        family inet;
        protocol direct;
        route-filter 0.0.0.0/0 prefix-length-range /32-/32;
    }
    then accept;
}
policy-statement load-balance {
    term 1 {
        then {
            load-balance per-packet;
        }
    }
}
community comm-VS_VLAN100 members target:1:100;
community comm-VS_VLAN200 members target:1:200;
community comm-VS_VLAN300 members target:1:300;
community comm-VS_VLAN400 members target:1:400;
community comm-leaf_esi members target:9999:9999;
```

Verify the configuration of the IRB interface.

```
user@core-2> show configuration interfaces irb
```

```
unit 1100 {
    family inet {
        address 172.16.0.4/24 {
            virtual-gateway-address 100.0.0.1;
        }
    }
}
unit 1200 {
    family inet {
        address 172.16.0.5/24 {
            virtual-gateway-address 200.0.0.1;
        }
    }
}
unit 1300 {
    family inet {
```

```
        address 10.10.10.3/24 {
            virtual-gateway-address 10.10.10.1;
        }
    }
}
unit 1400 {
    family inet {
        address 10.10.10.3/24 {
            virtual-gateway-address 10.10.10.1;
        }
    }
}
```

Verify the configuration of the EVPN-VXLAN BGP group.

```
user@core-2> show configuration protocols bgp group EVPN_VXLAN
```

```
type external;
local-address 10.255.255.1;
family evpn {
    signaling;
}
peer-as 65402;
multipath;
neighbor 192.0.2.2 {
    description leaf-1;
    multihop {
        ttl 255;
    }
}
neighbor 10.255.255.5 {
    description leaf-2;
    multihop {
        ttl 255;
    }
}
```

Verifying MAC Reachability to a Single-Homed CE Device (Leaf-1)

Purpose Verify MAC reachability to Tenant_C single-homed to Leaf-1. First you need to verify that the MAC address is learned locally on Leaf-1. Leaf-1 generates the Type-2 route only after it has learned the MAC address.

Action Verify that the MAC address is learned locally on Leaf-1.

```
lab@leaf-1> show ethernet-switching table vlan-id 300

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 : 5 entries, 5 learned
Routing instance : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v300                00:00:5e:00:01:01   DR       esi.1726
05:00:00:ff:78:00:00:05:14:00
   v300                00:21:59:c8:24:65   D        xe-0/0/34.0
   v300                00:21:59:c8:24:69   D        vtep.32771
10.255.255.5
   v300                40:a6:77:9a:43:f0   D        vtep.32769
10.255.255.0
   v300                40:a6:77:9a:47:f0   D        vtep.32770
10.255.255.1
```

Meaning The output shows that MAC 00:21:59:c8:24:65 is successfully learned towards the Tenant_C CE device on xe-0/0/34.0, and from the output below, we can see that we generate the Type-2 route to Core-1.

Verifying MAC Reachability to a Single-Homed CE Device (Type-2)

Purpose Verify MAC reachability to a single-homed CE device (Type-2)

Action Verify the generation of the Type-2 route to Core-1.

```
lab@leaf-1> show route advertising-protocol bgp 10.255.255.0 evpn-mac-address
00:21:59:c8:24:65

bgp.evpn.0: 77 destinations, 77 routes (77 active, 0 holddown, 0 hidden)
  Prefix    Nexthop        MED     Lclpref    AS path
  2:192.0.2.2:1::1300::00:21:59:c8:24:65/304
*                         Self                                    I
  2:192.0.2.2:1::1400::00:21:59:c8:24:65/304
*                         Self                                    I

default-switch.evpn.0: 67 destinations, 67 routes (67 active, 0 holddown, 0 hidden)

  Prefix    Nexthop        MED     Lclpref    AS path
  2:192.0.2.2:1::1300::00:21:59:c8:24:65/304
*                         Self                                    I
  2:192.0.2.2:1::1400::00:21:59:c8:24:65/304
*                         Self                                    I

__default_evpn__.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
```

Meaning The output shows that the same MAC address is advertised twice, with two different route targets. On xe-0/0/34, Tenant_C and Tenant_D are on the same physical server, running on different virtual machines. Tenant isolation is preserved by advertising these MACs on different VNIs and RTs.

On Core-1, the Type-2 route is received into bgp.evpn.0.

```
lab@core-1> show route receive-protocol bgp 192.0.2.2 evpn-mac-address
00:21:59:c8:24:65 extensive table bgp.evpn.0

[output omitted]
* 2:192.0.2.2:1::1300::00:21:59:c8:24:65/304 (2 entries, 0 announced)
     Import Accepted
     Route Distinguisher: 192.0.2.2:1
     Nexthop: 192.0.2.2
     AS path: 65402 I
     Communities: target:1:300 encapsulation0:0:0:0:vxlan

* 2:192.0.2.2:1::1400::00:21:59:c8:24:65/304 (2 entries, 0 announced)
     Import Accepted
     Route Distinguisher: 192.0.2.2:1
     Nexthop: 192.0.2.2
     AS path: 65402 I
     Communities: target:1:400 encapsulation0:0:0:0:vxlan
```

The output shows two Type-2 routes for 00:21:59:c8:24:65, but with different route targets. The route distinguisher is from Leaf-1, set as 192.0.2.2:1.

Verifying the Imported Route

Purpose Verify that the Type-2 route is imported.

Action On Core-1, verify whether this Type-2 route is successfully imported from the bgp.evpn.0 table into the EVPN switch instance.

Meaning The output shows that, in Tenant_C's virtual switch, the Type-2 route is advertised with the correct target, target:1:1300.

```
lab@core-1> show route table VS_VLAN300.evpn.0 evpn-mac-address 00:21:59:c8:24:65
 | grep 00:21:59:c8:24:65
2:192.0.2.2:1::1300::00:21:59:c8:24:65/304
```

You can use the **extensive** option to review the Type-2 route in greater detail.

```
lab@core-1> show route table VS_VLAN300.evpn.0 evpn-mac-address 00:21:59:c8:24:65
 extensive
```

```
[output omitted]
2:192.0.2.2:1::1300::00:21:59:c8:24:65/304 (2 entries, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 192.0.2.2:1
                Next hop type: Indirect
                Address: 0x2cf479c
                Next-hop reference count: 76
                Source: 192.0.2.2
                Protocol next hop: 192.0.2.2
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: [Secondary Active Ext]
                Local AS: 65400 Peer AS: 65402
                Age: 4:47  Metric2: 0
                Validation State: unverified
                Task: BGP_65402.192.0.2.2+179
                Announcement bits (1): 0-VS_VLAN300-evpn
                AS path: 65402 I
                Communities: target:1:300 encapsulation0:0:0:0:vxlan
                Import Accepted
                Localpref: 100
                Router ID: 192.0.2.2
                Primary Routing Table bgp.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 192.0.2.2
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 2
                                Next hop type: Router
                                Next hop: 10.0.0.1 via ge-1/0/0.0
                                Session Id: 0x140
                                Next hop: 10.0.0.3 via ge-1/0/1.0
                                Session Id: 0x141
   192.0.2.2/24 Originating RIB: inet.0
     Node path count: 1
     Forwarding nexthops: 2
    Nexthop: 10.0.0.1 via ge-1/0/0.0
         BGP    Preference: 170/-101
                Route Distinguisher: 192.0.2.2:1
                Next hop type: Indirect
                Address: 0x2cf479c
                Next-hop reference count: 76
                Source: 10.255.255.5
                Protocol next hop: 192.0.2.2
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: [Secondary NotBest Ext]
                Inactive reason: Not Best in its group - Router ID
                Local AS: 65400 Peer AS: 65402
                Age: 4:47  Metric2: 0
                Validation State: unverified
                Task: BGP_65402.10.255.255.5+61407
                AS path: 65402 I
                Communities: target:1:300 encapsulation0:0:0:0:vxlan
                Import Accepted
                Localpref: 100
                Router ID: 10.255.255.5
                Primary Routing Table bgp.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 192.0.2.2
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 2
```

```
                              Next hop type: Router
                              Next hop: 10.0.0.1 via ge-1/0/0.0
                              Session Id: 0x140
                              Next hop: 10.0.0.3 via ge-1/0/1.0
                              Session Id: 0x141
  192.0.2.2/24 Originating RIB: inet.0
    Node path count: 1
    Forwarding nexthops: 2
   Nexthop: 10.0.0.1 via ge-1/0/0.0
```

The output shows that Core-1 receives two copies. The first is the direct advertisement from Leaf-1 (Source: 192.0.2.2). The second is the indirect advertisement from Leaf-1 -> Leaf-2 -> Core-1 (Source: 10.255.255.5). Because we configured **no-nexthop-change** on Leaf-2, the protocol next hop of 192.0.2.2 is correct.

Verifying the Layer 2 Address Learning Daemon Copy

Purpose Verify the Layer 2 address learning daemon copy.

Action Verify the Layer 2 address learning daemon copy by entering the **show bridge-mac table** command.

Meaning The output shows that 00:21:59:c8:24:65 is reachable through the vtep.32774 logical interface to Leaf-1.

```
lab@core-1> show bridge mac-table instance VS_VLAN300

MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
    O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC)

Routing instance : VS_VLAN300
 Bridging domain : bd1300, VLAN : 300
   MAC                 MAC      Logical                Active
   address             flags    interface              source

   00:21:59:c8:24:65   D        vtep.32774             192.0.2.2

   00:21:59:c8:24:69   D        vtep.32783             10.255.255.5
```

NOTE: On EX9200 switches, the **show ethernet-switching table-instance** *instance-name* command corresponds to the **show bridge mac-table instance** *instance-name* command on MX Series routers

Verifying the Kernel-Level Forwarding Table

Purpose Verify the kernel-level forwarding table, next hop identifier, and Layer 2 MAC table and hardware.

Action Query the kernel-level forwarding table, correlate the index next hop identifier with the correct virtual network identifier, and review the Layer 2 MAC table and hardware.

Meaning Tenant_C's MAC, 00:21:59:c8:24:65, is reachable through index 699.

```
lab@core-1> show route forwarding-table family bridge vpn VS_VLAN300
Routing table: VS_VLAN300.evpn-vxlan
VPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      540     1
vtep.32774         intf     0                    comp      699     7
vtep.32783         intf     0                    comp      714     5

Routing table: VS_VLAN300.evpn-vxlan
Bridging domain: bd1300.evpn-vxlan
VPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
00:21:59:c8:24:65/48 user     0                  comp      699     7
00:21:59:c8:24:69/48 user     0                  comp      714     5
0x30003/51         user     0                    comp      706     2
```

Correlate index 699 (NH-Id) with the correct virtual network identifier 1300 and remote VTEP-ID of 192.0.2.2.

```
lab@core-1> show l2-learning vxlan-tunnel-end-point remote
Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
[default]
 RVTEP-IP         IFL-Idx   NH-Id
 192.0.2.2      351       697
    VNID          MC-Group-IP
    1100          0.0.0.0
 RVTEP-IP         IFL-Idx   NH-Id
 10.255.255.5      356       711
    VNID          MC-Group-IP
    1100          0.0.0.0
 RVTEP-IP         IFL-Idx   NH-Id
 192.0.2.2      352       698
    VNID          MC-Group-IP
    1200          0.0.0.0
 RVTEP-IP         IFL-Idx   NH-Id
 10.255.255.5      355       710
    VNID          MC-Group-IP
    1200          0.0.0.0
 RVTEP-IP         IFL-Idx   NH-Id
 192.0.2.2      353       699
    VNID          MC-Group-IP
    1300          0.0.0.0
 RVTEP-IP         IFL-Idx   NH-Id
```

```
10.255.255.5      357       714
   VNID          MC-Group-IP
   1300          0.0.0.0
RVTEP-IP         IFL-Idx   NH-Id
192.0.2.2      354       700
   VNID          MC-Group-IP
   1400          0.0.0.0
RVTEP-IP         IFL-Idx   NH-Id
10.255.255.5      358       709
   VNID          MC-Group-IP
   1400          0.0.0.0
```

NOTE: On EX9200 switches, the show ethernet-switching command corresponds to the show l2-learning command on MX Series routers.

The output shows that 00:21:59:c8:24:65 is successfully programmed in Packet Forwarding Engine hardware.

```
# show l2 manager mac-table
[output omitted]
route table name   : VS_VLAN300.7
  mac counters
    maximum   count
    0           2
  mac table information
  mac address       BD     learn  Entry  entry  hal    hardware info
                    Index  vlan   Flags  ifl    ifl    pfe  mask  ifl
  ------------------------------------------------------------------------
  00:21:59:c8:24:65  4     0     0x0014 vtep.32774 vtep.32774  0   -D    src
unknown dest vtep.32774
  00:21:59:c8:24:69  4     0     0x0014 vtep.32783 vtep.32783  0   -D    src
unknown dest vtep.32783
 Displayed 2 entries for routing instance VS_VLAN300.7
```

NOTE: On EX9200 switches, the show ethernet-switching command corresponds to the show l2-learning command on MX Series routers.

Verifying MAC Reachability to a Multihomed CE Device

Purpose Verify MAC reachability to the multihomed Tenant_B CE device on Leaf-1 and Leaf-2.

Action Verify that Leaf-1 and Leaf-2 are advertising both Type-1 and Type-2 reachability towards the multihomed CE device.

```
lab@leaf-1> show ethernet-switching table vlan-id 200

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 : 4 entries, 4 learned
Routing instance : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v200                00:00:5e:00:01:01   DR       esi.1727
05:00:00:ff:78:00:00:04:b0:00
   v200                00:21:59:c8:24:64   DL       ae1.0
   v200                40:a6:77:9a:43:f0   D        vtep.32769
10.255.255.2
   v200                40:a6:77:9a:47:f0   D        vtep.32770
10.255.255.1

lab@leaf-2> show ethernet-switching table vlan-id 200

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 : 5 entries, 5 learned
Routing instance : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v200                00:00:5e:00:01:01   DR       esi.1727
05:00:00:ff:78:00:00:04:b0:00
   v200                00:21:59:c8:24:41   DL       ae1.0
   v200                00:21:59:c8:24:68   DL       ae1.0
   v200                40:a6:77:9a:43:f0   D        vtep.32770
10.255.255.2
   v200                40:a6:77:9a:47:f0   D        vtep.32769
10.255.255.1
```

Verify that interface ae1 belongs on ESI 00:02:02:02:02:02:02:02:02:02.

```
lab@leaf-1> show configuration interfaces ae1 esi
00:02:02:02:02:02:02:02:02:02;
all-active;
```

Meaning The output shows that 00:21:59:c8:24:64 represents the physical MAC of the Tenant_B NIC physically attached to Leaf-1. 00:21:59:c8:24:68 represents the physical MAC of the Tenant_B NIC physically attached to Leaf-2. 00:21:59:c8:24:41 is a virtual MAC that, at the time of the capture, has been learned only on Leaf-2.

Verifying EVPN, Layer 2 Address Learning Daemon, and the Kernel-Forwarding Tables

Purpose Verify the Tenant B's EVPN table, and Core-1's Layer 2 address learning daemon table and kernel-forwarding table.

Action In Core-1, display the Tenant B's EVPN table.

```
lab@core-1> show route table VS_VLAN200.evpn.0

VS_VLAN200.evpn.0: 18 destinations, 31 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:192.0.2.2:0::010101010101010101::FFFF:FFFF/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                     [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                     > to 10.0.0.3 via ge-1/0/1.0
1:192.0.2.2:0::020202020202020202::FFFF:FFFF/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0

                     [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                     > to 10.0.0.3 via ge-1/0/1.0
1:192.0.2.2:1::010101010101010101::0/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                     > to 10.0.0.3 via ge-1/0/1.0
                     [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                     > to 10.0.0.3 via ge-1/0/1.0
1:192.0.2.2:1::020202020202020202::0/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                     > to 10.0.0.3 via ge-1/0/1.0
                     [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                     > to 10.0.0.3 via ge-1/0/1.0
```

```
1:10.255.255.5:0::010101010101010101::FFFF:FFFF/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
1:10.255.255.5:0::020202020202020202::FFFF:FFFF/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
1:10.255.255.5:1::010101010101010101::0/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
1:10.255.255.5:1::020202020202020202::0/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
[output omitted]
2:192.0.2.2:1::1200::00:21:59:c8:24:64/304
                   *[BGP/170] 4d 10:47:09, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 1d 00:44:04, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
2:10.255.255.5:1::1200::00:21:59:c8:24:41/304
                   *[BGP/170] 00:14:09, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 00:14:09, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
2:10.255.255.5:1::1200::00:21:59:c8:24:68/304
                   *[BGP/170] 1d 00:44:04, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
```

```
                    > to 10.0.0.3 via ge-1/0/1.0
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
1:10.255.255.5:1::020202020202020202::0/304
                   *[BGP/170] 23:27:33, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 23:27:33, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
2:192.0.2.2:1::1200::00:21:59:c8:24:64/304
                   *[BGP/170] 4d 10:47:09, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 1d 00:44:04, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
2:10.255.255.5:1::1200::00:21:59:c8:24:41/304
                   *[BGP/170] 00:14:09, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 00:14:09, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                    > to 10.0.0.1 via ge-1/0/0.0
                      to 10.0.0.3 via ge-1/0/1.0
2:10.255.255.5:1::1200::00:21:59:c8:24:68/304
                   *[BGP/170] 1d 00:44:04, localpref 100, from 192.0.2.2
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
                    [BGP/170] 4d 10:47:09, localpref 100, from 10.255.255.5
                      AS path: 65402 I, validation-state: unverified
                      to 10.0.0.1 via ge-1/0/0.0
                    > to 10.0.0.3 via ge-1/0/1.0
[output omitted]
```

Display Core-1's Layer 2 address learning daemon table.

```
lab@core-1> show bridge mac-table instance VS_VLAN200

MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
    O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC)

Routing instance : VS_VLAN200
```

```
Bridging domain : bd1200, VLAN : 200
  MAC                MAC      Logical              Active
  address            flags    interface            source

  00:21:59:c8:24:41  DR       esi.703
00:02:02:02:02:02:02:02:02:02
  00:21:59:c8:24:64  DR       esi.703
00:02:02:02:02:02:02:02:02:02
  00:21:59:c8:24:68  DR       esi.703
00:02:02:02:02:02:02:02:02:02
```

NOTE: On EX9200 switches, the **show ethernet-switching table-instance** ***instance-name*** command corresponds to the **show bridge mac-table instance** ***instance-name*** command on MX Series routers

Display Core-1's kernel forwarding table.

```
lab@core-1> show route forwarding-table vpn VS_VLAN200
Routing table: VS_VLAN200.evpn-vxlan
VPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      536     1
vtep.32774         intf     0                    comp      702     6
vtep.32778         intf     0                    comp      720     6

Routing table: VS_VLAN200.evpn-vxlan
Bridging domain: bd1200.evpn-vxlan
VPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
00:21:59:c8:24:41/48 user     0                  indr  1048579     4
                                                 comp      703     2
00:21:59:c8:24:64/48 user     0                  indr  1048579     4
                                                 comp      703     2
00:21:59:c8:24:68/48 user     0                  indr  1048579     4
                                                 comp      703     2
0x30002/51         user     0                    comp      714     2
```

Meaning For the Tenant_B CE device, four different routes are listed for ESI 00:02:02:02:02:02:02:02:02:02.

- 1:192.0.2.2:0::020202020202020202::FFFF:FFFF/304

  This per-Ethernet Segment A-D Type-1 EVPN route originated from Leaf-1. The route distinguisher is taken from global-level **routing-options**. Core-1 receives this Type-1 route, originated from Leaf-1, from both Leaf-1 and Leaf-2.

- 1:192.0.2.2:1::020202020202020202::0/304

This is the per-EVI A-D Type-1 EVPN route. The route distinguisher is taken from the routing instance, or in the case of QFX5100, **switch-options**. Core-1 receives this Type-1 route, originated from Leaf-1, from both Leaf-1 and Leaf-2.

- 1:10.255.255.5:0::02020202020202020::FFFF:FFFF/304

  This is the per-Ethernet Segment A-D Type-1 EVPN route originated from Leaf-2. The route distinguisher is taken from global-level **routing-options**. Core-1 receives this Type-1 route, originated from Leaf-2, from both Leaf-2 and Leaf-1.

- 1:10.255.255.5:1::02020202020202020::0/304

  This is the per-EVI A-D Type-1 EVPN route. The route distinguisher is taken from the routing instance, or in the case of QFX5100, **switch-options**. Core-1 receives this Type-1 route, originated from Leaf-2, from both Leaf-2 and Leaf-1.

Type-2 routes for the two physical and one virtual MAC belonging to the Tenant_B CE device originated as expected.

From the output we cannot yet determine what VTEPs are being used to forward to ESI 00:02:02:02:02:02:02:02:02:02. To determine the VTEPS, list the VXLAN tunnel endpoint ESIs.

```
lab@core-1> show l2-learning vxlan-tunnel-end-point esi

ESI                             RTT                       VLNBH INH     ESI-IFL
LOC-IFL   #RVTEPs
00:01:01:01:01:01:01:01:01:01 VS_VLAN200                  704   1048580 esi.704
          2
    RVTEP-IP             RVTEP-IFL      VENH     MASK-ID   FLAGS
    10.255.255.5          vtep.32778     720      1         2
    192.0.2.2           vtep.32774     702      0         2
[output omitted]
```

NOTE: On EX9200 switches, the **show ethernet-switching** command corresponds to the **show l2-learning** command on MX Series routers.

The output shows active load-balancing on the VTEP interfaces to both Leaf-1 and Leaf-2 for MACs on this ESI, which validates the all-active configuration on Leaf-1 and Leaf-2.

The reference to esi.703 might be confusing. This is an internal unilist next hop that comprises multiple, valid, individual ECMP next hops. You can correlate these individual next hops to the unilist by executing this shell level command:

```
lab@core-1> start shell command "nhinfo -di 703"
[output omitted]
NHs in list: 720, 702,
[output omitted]
```

From the output we can reference index 720 with vtep.32778 to Leaf-2 and index 702 with vtep.32774 to Leaf-1.

Verifying an EVPN Anycast Gateway

Purpose Verifying the EVPN anycast gateway is similar to the multihomed CE device scenario. For this example, verify from the leaf (QFX5100) perspective.

Action For this example, verify the anycast gateway on both Core-1 and Core-2 for VNI 1100 (Vlan 100 on both Leaf-1 and Leaf-2).

```
lab@leaf-1> show route receive-protocol bgp 10.255.255.2

inet.0: 13 destinations, 18 routes (13 active, 0 holddown, 0 hidden)

:vxlan.inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)

bgp.evpn.0: 75 destinations, 123 routes (75 active, 0 holddown, 0 hidden)
  Prefix    Nexthop        MED     Lclpref    AS path
  1:10.255.255.2:0::050000ff780000044c00::FFFF:FFFF/304
*                         10.255.255.2                                65400 I
  1:10.255.255.2:0::050000ff78000004b000::FFFF:FFFF/304
*                         10.255.255.2                                65400 I
  1:10.255.255.2:0::050000ff780000051400::FFFF:FFFF/304
*                         10.255.255.2                                65400 I
  1:10.255.255.2:0::050000ff780000057800::FFFF:FFFF/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:300::1300::00:00:5e:00:01:01/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:300::1300::40:a6:77:9a:43:f0/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:100::1100::00:00:5e:00:01:01/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:100::1100::40:a6:77:9a:43:f0/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:400::1400::00:00:5e:00:01:01/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:400::1400::40:a6:77:9a:43:f0/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:200::1200::00:00:5e:00:01:01/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:200::1200::40:a6:77:9a:43:f0/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:300::1300::00:00:5e:00:01:01::10.10.10.1/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:300::1300::40:a6:77:9a:43:f0::10.10.10.2/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:100::1100::00:00:5e:00:01:01::100.0.0.1/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:100::1100::40:a6:77:9a:43:f0::172.16.0.2/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:400::1400::00:00:5e:00:01:01::10.10.10.1/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:400::1400::40:a6:77:9a:43:f0::10.10.10.2/304
```

```
*                         10.255.255.2                                65400 I
  2:10.255.255.2:200::1200::00:00:5e:00:01:01::200.0.0.1/304
*                         10.255.255.2                                65400 I
  2:10.255.255.2:200::1200::40:a6:77:9a:43:f0::172.16.0.3/304
*                         10.255.255.2                                65400 I
[output omitted]
```

Verify the same Type-1, same anycast Type-2, and different physical Type-2 (a.b.c.3) from Core-2.

```
lab@leaf-1> show route receive-protocol bgp 10.255.255.1

inet.0: 13 destinations, 18 routes (13 active, 0 holddown, 0 hidden)

:vxlan.inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)

bgp.evpn.0: 75 destinations, 123 routes (75 active, 0 holddown, 0 hidden)
  Prefix    Nexthop        MED     Lclpref    AS path
  1:10.255.255.1:0::050000ff780000044c00::FFFF:FFFF/304
*                         10.255.255.1                                65400 I
  1:10.255.255.1:0::050000ff78000004b000::FFFF:FFFF/304
*                         10.255.255.1                                65400 I
  1:10.255.255.1:0::050000ff780000051400::FFFF:FFFF/304
*                         10.255.255.1                                65400 I
  1:10.255.255.1:0::050000ff780000057800::FFFF:FFFF/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:300::1300::00:00:5e:00:01:01/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:300::1300::40:a6:77:9a:47:f0/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:100::1100::00:00:5e:00:01:01/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:100::1100::40:a6:77:9a:47:f0/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:400::1400::00:00:5e:00:01:01/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:400::1400::40:a6:77:9a:47:f0/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:200::1200::00:00:5e:00:01:01/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:200::1200::40:a6:77:9a:47:f0/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:300::1300::00:00:5e:00:01:01::10.10.10.1/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:300::1300::40:a6:77:9a:47:f0::10.10.10.3/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:100::1100::00:00:5e:00:01:01::100.0.0.1/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:100::1100::40:a6:77:9a:47:f0::172.16.0.4/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:400::1400::00:00:5e:00:01:01::10.10.10.1/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:400::1400::40:a6:77:9a:47:f0::10.10.10.3/304
*                         10.255.255.1                                65400 I
  2:10.255.255.1:200::1200::00:00:5e:00:01:01::200.0.0.1/304
*                         10.255.255.1                                65400 I
```

```
  2:10.255.255.1:200::1200::40:a6:77:9a:47:f0::172.16.0.5/304
[output omitted]
```

Similarly, if we look at the default-switch.evpn.0 table on leaf-1, we see for the ESI for VNI1110:

```
lab@leaf-1> show route table default-switch.evpn.0 evpn-esi-value
05:00:00:ff:78:00:00:04:4c:00

default-switch.evpn.0: 66 destinations, 114 routes (66 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.255.2:0::050000ff780000044c00::FFFF:FFFF/304
                   *[BGP/170] 00:23:37, localpref 100, from 10.255.255.2
                      AS path: 65400 I, validation-state: unverified
                    > to 10.0.0.8 via xe-0/0/2.0
                      to 10.0.0.12 via xe-0/0/4.0
                    [BGP/170] 00:09:29, localpref 100, from 10.255.255.5
                      AS path: 65400 I, validation-state: unverified
                    > to 10.0.0.8 via xe-0/0/2.0
                      to 10.0.0.12 via xe-0/0/4.0
1:10.255.255.1:0::050000ff780000044c00::FFFF:FFFF/304
                   *[BGP/170] 00:23:33, localpref 100, from 10.255.255.1
                      AS path: 65400 I, validation-state: unverified
                    > to 10.0.0.8 via xe-0/0/2.0
                      to 10.0.0.12 via xe-0/0/4.0
                    [BGP/170] 00:09:29, localpref 100, from 10.255.255.5
                      AS path: 65400 I, validation-state: unverified
                    > to 10.0.0.8 via xe-0/0/2.0
                      to 10.0.0.12 via xe-0/0/4.0
```

Meaning The output shows that Leaf-1 receives Type-1 advertisements for the autogenerated ESI's for the anycast gateway. The outputs also show the anycast and physical MACs (a.b.c.2) for each bridge domain.

Two unique advertisements are received for this ESI. One from Core-1, and the other from Core-2, with corresponding route distinguishers. Because Leaf-1 is iBGP peered with Leaf-2, the extra copy for each route from Leaf-2 displays with the original protocol-next hop preserved.

Leaf-1's Layer 2 address learning daemon table displays:

```
lab@leaf-1> show ethernet-switching table vlan-id 100

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 : 5 entries, 5 learned
Routing instance : default-switch
   Vlan                MAC                 MAC      Logical                Active

   name                address             flags    interface              source

   v100                00:00:5e:00:01:01   DR       esi.1727
05:00:00:ff:78:00:00:04:4c:00
   v100                00:21:59:c8:24:63   DL       ae0.0
   v100                00:21:59:c8:24:69   D        vtep.32771
10.255.255.5
   v100                40:a6:77:9a:43:f0   D        vtep.32769
10.255.255.2
   v100                40:a6:77:9a:47:f0   D        vtep.32770
10.255.255.1
```

Leaf-1's kernel table displays:

```
lab@leaf-1> show route forwarding-table family ethernet-switching
[output omitted]

Routing table: default-switch.bridge
VPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd     1688     1
vtep.32769         intf     0                    comp     1724    19
vtep.32770         intf     0                    comp     1737    15
vtep.32771         intf     0                    comp     1738     9
[output omitted]

outing table: default-switch.bridge
Bridging domain: v100.bridge
VPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
00:00:5e:00:01:01/48 user     0                  comp     1724    19
[output omitted]
```

From the output, Leaf-1 has selected index 1724, correlating with vtep.32769, to forward all traffic for the anycast gateway on VNI 1100, and vtep.32769 is the VTEP to Core-1.

```
lab@leaf-1> show ethernet-switching vxlan-tunnel-end-point esi
[output omitted]

05:00:00:ff:78:00:00:04:4c:00 default-switch           1727  131074  esi.1727
         2
    RVTEP-IP             RVTEP-IFL      VENH     MASK-ID   FLAGS
    10.255.255.1          vtep.32770     1737     1         2
    10.255.255.2          vtep.32769     1724     0         2
```

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247

CHAPTER 12

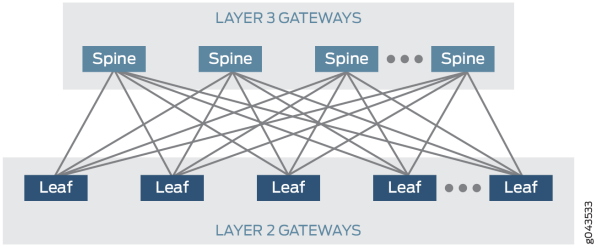
Configuring EVPN-VXLAN in a Topology With a Collapsed IP Fabric

Example: Configuring EVPN-VXLAN In a Collapsed IP Fabric Topology Within a Data Center

Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical servers and virtual machines [VMs]) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network. Virtual Extensible LAN (VXLAN) is a tunneling protocol that creates the data plane for the Layer 2 overlay network.

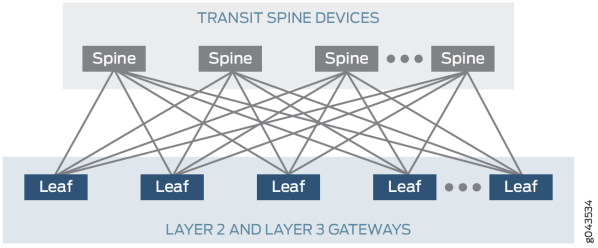
The physical underlay network over which EVPN-VXLAN is commonly deployed is a two-layer IP fabric, which includes spine and leaf devices as shown in Figure 30 on page 354. The spine devices—for example, QFX10000 switches—provide connectivity between the leaf devices, and the leaf devices—for example, QFX5100 switches—provide connectivity to attached hosts. In the overlay network, the leaf devices function as Layer 2 gateways that handle traffic within a VXLAN, and the spine devices function as Layer 3 gateways that handle traffic between VXLANs through the use of integrated routing and bridging (IRB) interfaces. For more information about configuring EVPN-VXLAN in a two-layer IP fabric, see "Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center" on page 302.

Figure 32: Two-Layer IP Fabric



You can also deploy EVPN-VXLAN over a physical underlay network in which the IP fabric is collapsed into a single layer of QFX10000 switches that function as leaf devices. In this collapsed fabric, which is shown in Figure 33 on page 452, the leaf devices serve as both Layer 2 and Layer 3 gateways. In this topology, transit spine devices provide Layer 3 routing functionality only.

Figure 33: Collapsed IP Fabric



This example describes how to configure EVPN-VXLAN, in particular, the Layer 3 gateway, on a leaf device in a collapsed IP fabric topology.

Requirements

This example uses the following hardware and software components:

- Two routers that function as transit spine devices.
- Three QFX10000 switches running Junos OS Release 15.1X53-D60 or later software. These switches are leaf devices that provide both Layer 2 and Layer 3 gateway functionality.

  NOTE: This example focuses on the configuration of the Layer 2 overlay network on a leaf device. The transit spine devices used in this example provide Layer 3 functionality only. As a result, this example does not include the configuration of these devices.

  Further, this example provides the configuration for leaf 1 only. The configuration for leaf 1 essentially serves as a template for the configuration of the other leaf devices. For the configuration of the other leaf devices, where appropriate, you can replace leaf 1-specific information with the information specific to the device you are configuring, add additional commands, and so on.

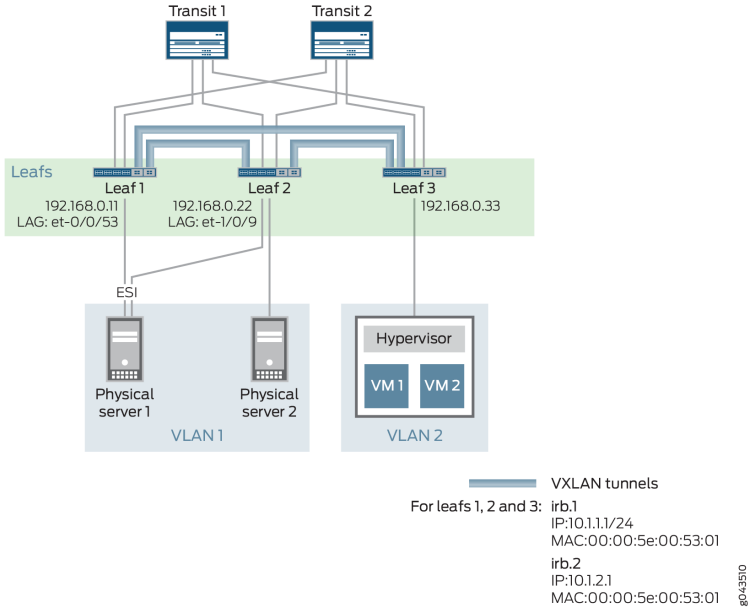
- Two physical (bare-metal) servers and one server with VMs that are supported by a hypervisor.

Overview and Topology

The collapsed IP fabric topology shown in Figure 34 on page 453 includes two transit spine devices, a collapsed IP fabric, which includes three leaf devices that function as both Layer 2 and Layer 3 gateways, two physical servers, and one virtualized server on which VMs and a hypervisor are installed. Physical server 1 is connected to leaf 1 and leaf 2 through a link aggregation group (LAG) interface. On both leaf devices, the interface is assigned the same Ethernet segment identifier (ESI) and set to multihoming active-active mode.

All leaf devices are in the same autonomous system (65200).

Figure 34: Collapsed IP Fabric Topology Within a Data Center



In this topology, an application on physical server 1 needs to communicate with VM 1 on the virtualized server. Physical servers 1 and 2 are included in VLAN 1, and the virtualized server is included in VLAN 2. For communication between VLANs 1 and 2 to occur, two IRB interfaces—irb.1, which is associated with VLAN 1, and irb.2, which is associated with VLAN 2—must be configured on each leaf device.

The most significant difference between the configuration of an EVPN-VXLAN overlay network deployed over a collapsed IP fabric and the configuration of an overlay network deployed over a two-layer IP fabric is the configuration of the Layer 3 gateway. Therefore, this example focuses on the EVPN-VXLAN configuration, in particular, the Layer 3 gateway configuration on the leaf devices.

For the collapsed IP fabric topology, you can configure the IRB interfaces within an EVPN instance using one of the following methods:

- **Method 1**—For each IRB interface on a particular leaf device, for example, leaf 1, the following is specified:
  - A unique IP address.
  - The *same* MAC address.

  For example:

irb.1	IP address: 10.1.1.1/24	MAC address: 00:00:5e:00:53:01
irb.2	IP address: 10.1.2.1/24	MAC address: 00:00:5e:00:53:01

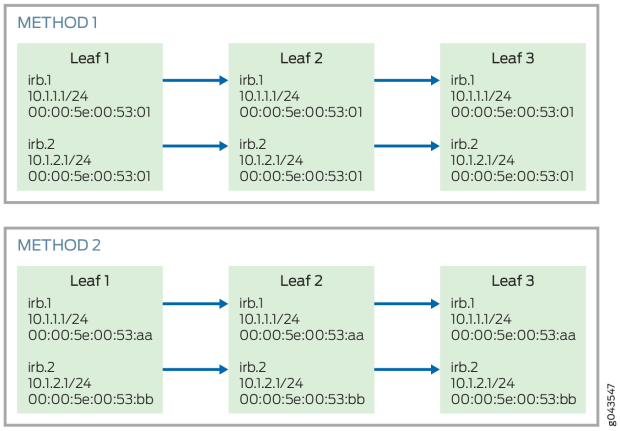
- **Method 2**—For each IRB interface on leaf 1, the following is specified:
  - A unique IP address.
  - A *unique* MAC address.

  For example:

irb.1	IP address: 10.1.1.1/24	MAC address: 00:00:5e:00:53:aa
irb.2	IP address: 10.1.2.1/24	MAC address: 00:00:5e:00:53:bb

Regardless of the method that you use to configure the IRB interfaces on leaf 1, if irb.1 and irb.2 are also configured on other leafs in the collapsed IP fabric, for example, leafs 2 and 3, you must specify the same configurations that you specified on leaf 1 for those IRB interfaces on leafs 2 and 3. For example, Figure 35 on page 454 shows the configurations for irb.1 and irb.2 on leafs 1, 2, and 3 for both methods.

Figure 35: Method 1 and 2 IRB Interface Configurations on Multiple Leaf Devices



NOTE: In this example, method 1 is used to configure the IRB interfaces.

As shown in this example, with the same MAC address configured for each IRB interface on each leaf device, each host uses the same MAC address when sending inter-VLAN traffic regardless of where the host is located or which leaf device receives the traffic. For example, in the topology shown in Figure 34 on page 453, multi-homed physical server 1 in VLAN 1 sends a packet to VM 1 in VLAN 2. If leaf 1 is down, leaf 2 continues to forward the inter-VLAN traffic even without the configuration of a redundant default gateway MAC address.

Note that the configuration of IRB interfaces used in this example does not include a virtual gateway address (VGA) and a corresponding MAC address that establishes redundant default gateway functionality, which is mentioned above. By configuring the same MAC address for each IRB interface on each leaf device, hosts use the local leaf device configured with the common MAC address as the default Layer 3 gateway. Therefore, you eliminate the need to advertise a redundant default gateway and dynamically synchronize the MAC addresses of the redundant default gateway throughout the EVPN control plane. As a result, when configuring each leaf device, you must disable the advertisement of the redundant default gateway by including the **default-gateway do-not-advertise** configuration statement in the **[edit protocols evpn]** hierarchy level in your configuration.

NOTE: Although the IRB interface configuration used in this example does not include a VGA, you can configure it as needed to make EVPN-VXLAN work properly in your collapsed IP fabric topology. If you configure a VGA for each IRB interface, you specify the same IP address for each VGA on each leaf device instead of configuring the same MAC address for each IRB interface on each leaf device as is shown in this example.

When it comes to handling the replication of broadcast, unknown unicast, and multicast (BUM) traffic, note that the configuration on leaf 1:

- includes the **set protocols evpn multicast-mode ingress-replication** command. This command causes leaf 1, which is a hardware VTEP, to handle the replication and sending of BUM traffic instead of a multicast client in the EVPN-VXLAN topology.
- Retains the QFX10000 switch's default setting of disabled for ingress node replication for EVPN-VXLAN. With this feature disabled, if a QFX10000 switch that functions as a VTEP receives a BUM packet intended, for example, for a physical server in a VLAN with the VNI of 1001, the VTEP replicates and sends the packet only to VTEPs on which the VNI of 1001 is configured. If this feature is enabled, the VTEP replicates and sends this packet to all VTEPs in its database, including those that do not have VNI 1001 configured. To prevent a VTEP from needlessly flooding BUM traffic throughout an EVPN-VXLAN overlay network, we strongly recommend that if not already disabled, you disable ingress node replication on each of the leaf devices by specifying the **delete vlans *vlan-name* vxlan ingress-node-replication** command.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Leaf 1

```
set interfaces et-0/0/53 ether-options 802.3ad ae202
set interfaces ae202 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae202 esi all-active
set interfaces ae202 aggregated-ether-options lacp active
set interfaces ae202 aggregated-ether-options lacp system-id 00:00:00:04:04:04
set interfaces ae202 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae202 unit 0 family ethernet-switching vlan members 1-2
set interfaces irb unit 1 family inet address 10.1.1.1/24
set interfaces irb unit 1 mac 00:00:5e:00:53:01
set interfaces irb unit 2 family inet address 10.1.2.1/24
set interfaces irb unit 2 mac 00:00:5e:00:53:01
set interfaces lo0 unit 0 family inet address 192.168.0.11/32
set interfaces lo0 unit 1 family inet address 192.168.10.11/32
set protocols bgp group overlay-evpn type internal
set protocols bgp group overlay-evpn local-address 192.168.0.11
set protocols bgp group overlay-evpn family evpn signaling
set protocols bgp group overlay-evpn local-as 65200
set protocols bgp group overlay-evpn multipath
set protocols bgp group overlay-evpn neighbor 192.168.0.22
set protocols bgp group overlay-evpn neighbor 192.168.0.33
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list 1001
set protocols evpn extended- vni-list 1002
set protocols evpn multicast-mode ingress-replication
set protocols evpn default-gateway do-not-advertise
set protocols evpn vni-options vni 1001 vrf-target export target:1:1001
set protocols evpn vni-options vni 1002 vrf-target export target:1:1002
set policy-options community comm-leaf_esi members target 9999:9999
set policy-options community com1001 members target:1:1001
set policy-options community com1002 members target:1:1002
set policy-options policy-statement LEAF-IN term import_leaf_esi from community
  comm-leaf_esi
set policy-options policy-statement LEAF-IN term import_leaf_esi then accept
set policy-options policy-statement vrf-1-to-200 term import_vni1001 from community
  com1001
set policy-options policy-statement vrf-1-to-200 term import_vni1001 then accept
set policy-options policy-statement vrf-1-to-200 term import_vni1002 from community
  com1002
set policy-options policy-statement vrf-1-to-200 term import_vni1002 then accept
set routing-instances VRF_1 instance-type vrf
set routing-instances VRF_1 interface irb.1
set routing-instances VRF_1 interface irb.2
set routing-instances VRF_1 interface lo0.1
set routing-instances VRF_1 route-distinguisher 192.168.0.11:1
set routing-instances VRF_1 vrf-target target:1:1
set switch-options vtep-source-interface lo0.0
```

```
set switch-options route-distinguisher 192.168.0.11:5000
set switch-options vrf-import LEAF-IN
set switch-options vrf-import vrf-1-to-200
set switch-options vrf-target target:9999:9999
set vlans bd1 vlan-id 1
set vlans bd1 l3-interface irb.1
set vlans bd1 vxlan vni 1001
delete vlans bd1 vxlan ingress-node-replication
set vlans bd2 vlan-id 2
set vlans bd2 l3-interface irb.2
set vlans bd2 vxlan vni 1002
delete vlans bd2 vxlan ingress-node-replication
```

Configuring EVPN-VXLAN on Leaf 1

Step-by-Step Procedure

1. Enable physical server 1 to be multihomed to leaf 1 and leaf 2 by configuring an aggregated Ethernet interface, specifying an ESI for the interface, and setting the mode so that the connections to both leaf devices are active.

   NOTE: When configuring the ae202 interface on leaf 2, you must specify the same ESI (00:11:22:33:44:55:66:77:88:99) that is specified for the same interface on leaf 1.

   ```
   [edit]
   user@switch# set interfaces et-0/0/53 ether-options 802.3ad ae202
   user@switch# set interfaces ae202 esi 00:11:22:33:44:55:66:77:88:99
   user@switch# set interfaces ae202 esi all-active
   user@switch# set interfaces ae202 aggregated-ether-options lacp active
   user@switch# set interfaces ae202 aggregated-ether-options lacp system-id
     00:00:00:04:04:04
   user@switch# set interfaces ae202 unit 0 family ethernet-switching interface-mode
     trunk
   user@switch# set interfaces ae202 unit 0 family ethernet-switching vlan members
     1-2
   ```

2. Configure two IRB interfaces, each with unique IP addresses and the same MAC address.

   ```
   [edit]
   user@switch# set interfaces irb unit 1 family inet address 10.1.1.1/24
   user@switch# set interfaces irb unit 1 mac 00:00:5e:00:53:01
   user@switch# set interfaces irb unit 2 family inet address 10.1.2.1/24
   user@switch# set interfaces irb unit 2 mac 00:00:5e:00:53:01
   ```

3. Configure a loopback interface (lo0.0) for the leaf device and a logical loopback address (lo0.1) for the EVPN routing instance (VRF-1).

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 192.168.0.11/32
user@switch# set interfaces lo0 unit 1 family inet address 192.168.10.11/32
```

4. Set up the IBGP overlay network.

```
[edit]
user@switch# set protocols bgp group overlay-evpn type internal
user@switch# set protocols bgp group overlay-evpn local-address 192.168.0.11
user@switch# set protocols bgp group overlay-evpn family evpn signaling
user@switch# set protocols bgp group overlay-evpn local-as 65200
user@switch# set protocols bgp group overlay-evpn multipath
user@switch# set protocols bgp group overlay-evpn neighbor 192.168.0.22
user@switch# set protocols bgp group overlay-evpn neighbor 192.168.0.33
```

5. Set up the EVPN-VXLAN domain, which entails determining which VNIs are included in the domain, specifying that leaf 1, which is a hardware VTEP, handles the replication and sending of BUM traffic, disabling the advertisement of the redundant default gateway throughout the EVPN control plane, and specifying a route target for each VNI.

```
[edit]
user@switch# set protocols evpn encapsulation vxlan
user@switch# set protocols evpn extended-vni-list 1001
user@switch# set protocols evpn extended- vni-list 1002
user@switch# set protocols evpn multicast-mode ingress-replication
user@switch# set protocols evpn default-gateway do-not-advertise
user@switch# set protocols evpn vni-options vni 1001 vrf-target export target:1:1001
user@switch# set protocols evpn vni-options vni 1002 vrf-target export target:1:1002
```

6. Set up communities for the VNIs, and create policies that import and accept the overlay routes.

```
[edit]
user@switch# set policy-options community comm-leaf_esi members target
  9999:9999
user@switch# set policy-options community com1001 members target:1:1001
user@switch# set policy-options community com1002 members target:1:1002
user@switch# set policy-options policy-statement LEAF-IN term import_leaf_esi
  from community comm-leaf_esi
user@switch# set policy-options policy-statement LEAF-IN term import_leaf_esi
  then accept
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1001
  from community com1001
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1001
  then accept
```

```
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1002
  from community com1002
user@switch# set policy-options policy-statement vrf-1-to-200 term import_vni1002
  then accept
```

7. Set up an EVPN routing instance.

```
[edit]
user@switch# set routing-instances VRF_1 instance-type vrf
user@switch# set routing-instances VRF_1 interface irb.1
user@switch# set routing-instances VRF_1 interface irb.2
user@switch# set routing-instances VRF_1 interface lo0.1
user@switch# set routing-instances VRF_1 route-distinguisher 192.168.0.11:1
user@switch# set routing-instances VRF_1 vrf-target target:1:1
```

NOTE: In the above EVPN routing instance configuration, a unique logical loopback interface (lo0.1) is specified, and an IP address for the interface is specified using the set interfaces lo0 unit *logical-unit-number* family inet address *ip-address/prefix* command. All items configured in the above routing instance except for the logical loopback interface are required for EVPN. However, the configuration of a logical loopback interface and associated IP address are required to ensure that VXLAN control packets are properly processed.

8. Configure the switch options to use loopback interface lo0.0 as the source interface of the VTEP, set a route distinguisher, and import the route targets for the three communities into the EVPN (MAC) table.

```
[edit]
user@switch# set switch-options vtep-source-interface lo0.0
user@switch# set switch-options route-distinguisher 192.168.0.11:5000
user@switch# set switch-options vrf-import LEAF-IN
user@switch# set switch-options vrf-import vrf-1-to-200
user@switch# set switch-options vrf-target target:9999:9999
```

9. Configure VLANs to which IRB interfaces and VXLAN VNIs are associated.

```
[edit]
user@switch# set vlans bd1 vlan-id 1
user@switch# set vlans bd1 l3-interface irb.1
user@switch# set vlans bd1 vxlan vni 1001
user@switch# set vlans bd2 vlan-id 2
user@switch# set vlans bd2 l3-interface irb.2
user@switch# set vlans bd2 vxlan vni 1002
```

10. If not already disabled, disable ingress node replication to prevent leaf 1 from needlessly flooding BUM traffic throughout the EVPN-VXLAN overlay network.

```
[edit]
user@switch# delete vlans bd1 vxlan ingress-node-replication
user@switch# delete vlans bd2 vxlan ingress-node-replication
```

Verification

The section describes the following verifications for this example:

- Verifying the IRB Interfaces on page 460
- Verifying the VTEP Interfaces on page 460
- Verifying the EVPN Routing Instance on page 461

Verifying the IRB Interfaces

Purpose Verify that the IRB interfaces are up and running.

Action Display the status of the IRB interfaces:

```
user@leaf1> show interfaces irb terse

Interface               Admin Link Proto    Local                 Remote
irb                     up    up
irb.1                   up    up   inet     10.1.1.1/24
irb.2                   up    up   inet     10.1.2.1/24
```

Meaning The IRB interfaces are up and running.

Verifying the VTEP Interfaces

Purpose Verify the status of the VTEP interfaces.

Action Display the status of the VTEP interfaces:

```
user@leaf1> show interfaces vtep terse
```

```
Interface               Admin Link Proto    Local                 Remote
vtep                    up    up
vtep.32769              up    up   eth-switch
vtep.32770              up    up   eth-switch
vtep.32771              up    up   eth-switch
```

Meaning The interfaces for each of the VTEPs is up. Therefore, the VTEP interfaces are functioning normally.

Verifying the EVPN Routing Instance

Purpose Verify the routing table for VRF_1.

Action Verify the routing table for the EVPN routing instance VRF_1.

```
user@leaf1> show route table VRF_1.inet.0
```

```
VRF_1.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.0/24        *[Direct/0] 00:07:38
                    > via irb.1
10.1.1.1/32        *[Local/0] 00:07:38
                      Local via irb.110.1.2.0/24        *[Direct/0] 00:07:38
                    > via irb.2
10.1.2.1/32        *[Local/0] 00:07:38
                      Local via irb.2
192.168.10.11/32   *[Direct/0] 00:07:38
                    > via lo0.1
```

Meaning The EVPN routing instance is functioning correctly.

Example: Configuring a QFX5110 Switch as Layer 2 and 3 VXLAN Gateways in an EVPN-VXLAN Topology with a Collapsed IP Fabric

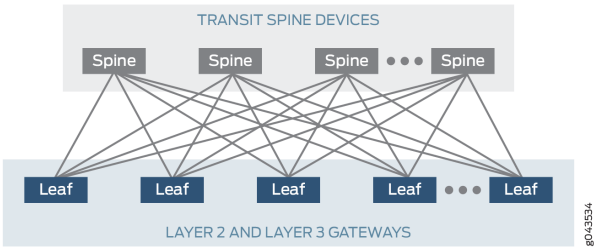
Ethernet VPN (EVPN) is a control plane technology that enables hosts (physical [bare-metal] servers and virtual machines [VMs]) to be placed anywhere in a network and remain connected to the same logical Layer 2 overlay network. Virtual Extensible LAN (VXLAN) is a tunneling protocol that creates the data plane for the Layer 2 overlay network.

You can deploy EVPN-VXLAN over a physical underlay network in which the IP fabric is collapsed into a single layer of QFX5110 switches that function as leaf devices. In this

collapsed fabric, which is shown in Figure 33 on page 452, the leaf devices serve as both Layer 2 and Layer 3 VXLAN gateways. In the EVPN-VXLAN overlay network, Layer 2 VXLAN gateways handle traffic within a VLAN, and Layer 3 VXLAN gateways handle traffic between VLANs using integrated routing and bridging (IRB) interfaces.

Figure 33 on page 452 also shows transit spine devices, which provide Layer 3 routing functionality only.

Figure 36: Collapsed IP Fabric



Starting with Junos OS Release 17.3R1, the QFX5110 switch can function as a leaf device, which acts as Layer 2 and 3 VXLAN gateways in an EVPN-VXLAN topology with a collapsed IP fabric.

This topic provides a sample configuration of a QFX5110 switch that functions as a leaf device in an EVPN-VXLAN topology with a collapsed IP fabric.

Requirements

This example uses the following hardware and software components:

- Two routers that function as transit spine devices.
- Three QFX5110 switches running Junos OS Release 17.3R1 or later. These switches act as leaf devices (leaf 1, leaf 2, and leaf 3) that provide Layer 2 and 3 VXLAN gateway functionality.

NOTE: This example focuses on the configuration of the QFX5110 switch that functions as leaf 1. A basic configuration is provided for the IP/BGP underlay network, the EVPN-VXLAN overlay network, a customer-specific profile, and route leaking. This example does not include all features that can be used in an EVPN-VXLAN network. The configuration for leaf 1 essentially serves as a template for the configuration of the other leaf devices. For the configuration of the other leaf devices, where appropriate, you can replace leaf 1-specific information with the information specific to the device you are configuring, add additional commands, and so on.

- Two physical servers and one virtualized server with VMs that are supported by a hypervisor.

Overview and Topology

In this example, a service provider supports ABC Corporation, which has multiple sites. Physical servers in site 100 must communicate with VMs in site 200. To enable this communication in the EVPN-VXLAN topology with a collapsed IP fabric shown in Figure 37 on page 463, you configure the key software entities in Table 17 on page 464 on the QFX5110 switches that function as Layer 2 and 3 VXLAN gateways, or leaf devices.

Figure 37: EVPN-VXLAN Topology with a Collapsed IP Fabric

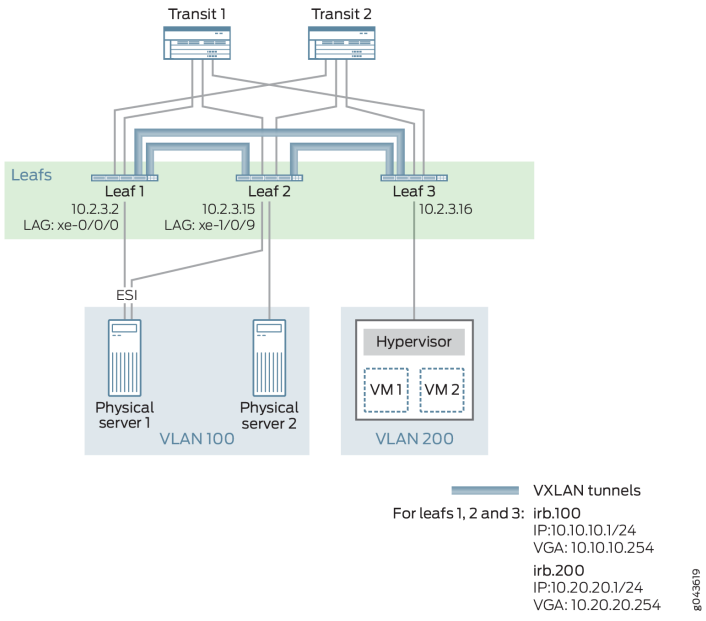


Table 17: Layer 3 Inter-VLAN Routing Entities Configured on Leaf 1, Leaf 2, and Leaf 3

Entities	Configuration on Leaf 1, Leaf 2, and Leaf 3
VLANs	v100 v200
VRF instances	vrf_vlan100 vrf_vlan200
IRB interfaces	irb.100 10.10.10.1/24 (IRB IP address) 10.10.10.254 (virtual gateway address)
	irb.200 10.20.20.1/24 (IRB IP address) 10.20.20.254 (virtual gateway address)

As outlined in Table 17 on page 464, you configure VLAN v100 for site 100 and VLAN v200 for site 200 on each leaf device. To segregate the Layer 3 routes for VLANs v100 and v200, you create VPN routing and forwarding (VRF) instances vrf_vlan100 and vrf_vlan200 on each leaf device. To route traffic between the VLANs, you configure IRB interfaces irb.100 and irb.200, and associate VRF instance vrf_vlan100 with IRB interface irb.100, and VRF instance vrf_vlan200 with IRB interface irb.200.

The physical servers in VLAN v100 are non-virtualized. As a result, we strongly recommend that you configure IRB interfaces irb.100 and irb.200 to function as default Layer 3 gateways that handle the inter-VLAN traffic of the physical servers. To that end, the configuration of each IRB interface also includes a virtual gateway address (VGA), which configures an IRB interface as a default Layer 3 gateway. In addition, this example assumes that each physical server is configured to use a particular default gateway. For more information about default gateways and how inter-VLAN traffic flows between a physical server to another physical server or VM in another VLAN in an EVPN-VXLAN topology with a collapsed IP fabric, see "Using a Default Layer 3 Gateway to Route Traffic Between Virtual Networks in an EVPN-VXLAN Topology" on page 293.

NOTE: When configuring a VGA for an IRB interface, keep in mind that the IRB IP address and VGA must be different.

NOTE: If a QFX5110 switch running Junos OS Release 17.3R1 or later functions as both a Layer 3 VXLAN gateway and a Dynamic Host Configuration Protocol (DHCP) relay in an EVPN-VXLAN topology, the DHCP server response time for an IP address might take up to a few minutes. The lengthy response time might occur if a DHCP client receives and later releases an IP address on an EVPN-VXLAN IRB interface configured on the QFX5110 switch and the binding between the DHCP client and the IP address is not deleted.

As outlined in Table 17 on page 464, a separate VRF routing instance is configured for each VLAN. To enable the communication between hosts in VLANs v100 and v200, this example shows how to export unicast routes from the routing table for vrf_vlan100 and import the routes into the routing table for vrf_vlan200 and vice versa. This feature is also known as route leaking.

Basic Underlay Network Configuration

CLI Quick Configuration

To quickly configure a basic underlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set routing-options router-id 10.2.3.2
set routing-options autonomous-system 64500
set protocols bgp group pe neighbor 10.2.3.15
set protocols bgp group pe neighbor 10.2.3.16
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface em0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Configuring the Underlay Network

Step-by-Step Procedure

To configure a basic underlay network on leaf 1:

1. Configure the router ID and autonomous system number for leaf 1.

   ```
   [edit routing-options]
   user@switch# set router-id 10.2.3.2
   user@switch# set autonomous-system 64500
   ```

2. Configure a BGP group that includes leaf 2 and leaf 3 as peers that also handle underlay functions.

   ```
   [edit protocols]
   user@switch# set bgp group pe neighbor 10.2.3.15
   user@switch# set bgp group pe neighbor 10.2.3.16
   ```

3. Configure OSPF as the routing protocol for the underlay network.

```
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface all
user@switch# set ospf area 0.0.0.0 interface em0.0 disable
user@switch# set ospf area 0.0.0.0 interface lo0.0 passive
```

Basic EVPN-VXLAN Overlay Network Configuration

CLI Quick Configuration

To quickly configure a basic overlay network, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set forwarding-options vxlan-routing interface-num 8192
set forwarding-options vxlan-routing next-hop 16384
set protocols bgp group pe type internal
set protocols bgp group pe local-address 10.2.3.2
set protocols bgp group pe family evpn signaling
set protocols evpn encapsulation vxlan
set protocols evpn extended-vni-list all
set protocols evpn default-gateway no-gateway-community
set switch-options route-distinguisher 10.2.3.2:1
set switch-options vrf-target target:1111:11
set switch-options vtep-source-interface lo0.0
```

Configuring a Basic EVPN-VXLAN Underlay Network

Step-by-Step Procedure

To configure a basic EVPN-VXLAN overlay network on leaf 1:

1. Increase the number of physical interfaces and next hops that the QFX5110 switch allocates for use in an EVPN-VXLAN topology.

```
[edit forwarding-options]
set vxlan-routing interface-num 8192
set vxlan-routing next-hop 16384
```

2. Configure an IBGP overlay between leaf 1 and the other two leaf devices, specify a local IP address for leaf 1, and include the EVPN signaling Network Layer Reachability Information (NLRI) to the BGP group.

```
[edit protocols]
user@switch# set bgp group pe type internal
user@switch# set bgp group pe local-address 10.2.3.2
user@switch# set bgp group pe family evpn signaling
```

3. Configure VXLAN encapsulation for the data packets exchanged between the EVPN neighbors, and specify that all VXLAN network identifiers (VNIs) are part of the virtual routing and forwarding (VRF) instance. Also, specify that the MAC address of the IRB interface and the MAC address of the corresponding default gateway are advertised without the extended community option of default -gateway.

```
[edit protocols]
user@switch# set evpn encapsulation vxlan
user@switch# set evpn extended-vni-list all
user@switch# set evpn default-gateway no-gateway-community
```

4. Configure switch options to set a route distinguisher and VRF target for the VRF routing instance, and associate interface lo0 with the virtual tunnel endpoint (VTEP).

```
[edit switch-options]
user@switch# set route-distinguisher 10.2.3.2:1
user@switch# set vrf-target target:1111:11
user@switch# set vtep-source-interface lo0.0
```

Basic Customer Profile Configuration

CLI Quick Configuration

To quickly configure a basic customer profile for ABC Corporation sites 100 and 200, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces xe-0/0/0 ether-options 802.3ad ae0
set interfaces ae0 esi 00:00:00:ab:cd:00:01:00:00:01
set interfaces ae0 esi all-active
set interfaces xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/10 unit 0 family ethernet-switching vlan members v100
set interfaces xe-0/0/11 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/11 unit 0 family ethernet-switching vlan members v200
set interfaces irb unit 100 family inet address 10.10.10.1/24 virtual-gateway-address
  10.10.10.254
set interfaces irb unit 200 family inet address 10.20.20.1/24 virtual-gateway-address
  10.20.20.254
set interfaces lo0 unit 0 family inet address 10.2.3.2/32 primary
set interfaces lo0 unit 1 family inet address 10.2.3.24/32 primary
set interfaces lo0 unit 2 family inet address 10.2.3.25/32 primary
set routing-instances vrf_vlan100 instance-type vrf
set routing-instances vrf_vlan100 interface irb.100
set routing-instances vrf_vlan100 interface lo0.1
set routing-instances vrf_vlan100 route-distinguisher 10.2.3.11:2
set routing-instances vrf_vlan200 instance-type vrf
set routing-instances vrf_vlan200 interface irb.200
set routing-instances vrf_vlan200 interface lo0.2
set routing-instances vrf_vlan200 route-distinguisher 10.2.3.11:3
set vlans v100 vlan-id 100
set vlans v100 l3-interface irb.100
```

```
set vlans v100 vxlan vni 100
set vlans v200 vlan-id 200
set vlans v200 l3-interface irb.200
set vlans v200 vxlan vni 200
```

Configuring a Basic Customer Profile

Step-by-Step Procedure

To configure a basic customer profile for ABC Corporation sites 100 and 200 on leaf 1:

1. Enable physical server 1 to be multihomed to leaf 1 and leaf 2 by configuring an aggregated Ethernet interface, specifying an ESI for the interface, and setting the mode so that the connections to both leaf devices are active.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/0 ether-options 802.3ad ae0
   user@switch# set ae0 esi 00:11:22:33:44:55:66:77:88:99
   user@switch# set ae0 esi all-active
   ```

   NOTE: When configuring the ae202 interface on leaf 2, you must specify the same ESI (00:11:22:33:44:55:66:77:88:99) that is specified for the same interface on leaf 1.

2. Configure Layer 2 interfaces, and specify each interface as a member of VLAN v100 or v200.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/10 unit 0 family ethernet-switching interface-mode trunk
   user@switch# set xe-0/0/10 unit 0 family ethernet-switching vlan members v100
   user@switch# set xe-0/0/11 unit 0 family ethernet-switching interface-mode trunk
   user@switch# set xe-0/0/11 unit 0 family ethernet-switching vlan members v200
   ```

3. Configure IRB interfaces and associated VGAs (default Layer 3 virtual gateways), which enable the communication between physical servers, or physical servers and VMs, in different VLANs.

   ```
   [edit interfaces]
   user@switch# set irb unit 100 family inet address 10.10.10.1/24
     virtual-gateway-address 10.10.10.254
   user@switch# set irb unit 200 family inet address 10.20.20.1/24
     virtual-gateway-address 10.20.20.254
   ```

   NOTE: When configuring a VGA for an IRB interface, keep in mind that the IRB IP address and VGA must be different.

4. Configure a loopback interface (lo0) for leaf 1 and a logical loopback address (lo0.*x*) for each VRF routing instance.

```
[edit interfaces]
user@switch# set lo0 unit 0 family inet address 10.2.3.2/32 primary
user@switch# set lo0 unit 1 family inet address 10.2.3.24/32 primary
user@switch# set lo0 unit 2 family inet address 10.2.3.25/32 primary
```

5. Configure a VRF routing instance for VLAN v100 and another VRF routing instance for VLAN v200. In each routing instance, associate an IRB interface, a loopback interface, and an identifier attached to the route.

```
[edit routing-instances]
user@switch# set vrf_vlan100 instance-type vrf
user@switch# set vrf_vlan100 interface irb.100
user@switch# set vrf_vlan100 interface lo0.1
user@switch# set vrf_vlan100 route-distinguisher 10.2.3.2:2
user@switch# set vrf_vlan200 instance-type vrf
user@switch# set vrf_vlan200 interface irb.200
user@switch# set vrf_vlan200 interface lo0.2
user@switch# set vrf_vlan200 route-distinguisher 10.2.3.2:3
```

6. Configure VLANs v100 and v200, and associate an IRB interface and VNI with each VLAN.

```
[edit vlans]
user@switch# set v100 vlan-id 100
user@switch# set v100 l3-interface irb.100
user@switch# set v100 vxlan vni 100
user@switch# set v200 vlan-id 200
user@switch# set v200 l3-interface irb.200
user@switch# set v200 vxlan vni 200
```

Route Leaking Configuration

CLI Quick Configuration

To quickly configure route leaking, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set policy-options policy-statement export-inet1 term 1 from interface irb.100
set policy-options policy-statement export-inet1 term 1 then community add com200
set policy-options policy-statement export-inet1 term 1 then accept
set policy-options policy-statement export-inet2 term 1 from interface irb.200
set policy-options policy-statement export-inet2 term 1 then community add com100
set policy-options policy-statement export-inet2 term 1 then accept
set policy-options policy-statement import-inet term 1 from community com100
set policy-options policy-statement import-inet term 1 from community com200
set policy-options policy-statement import-inet term 1 then accept
```

```
set policy-options community com100 members target:1:100
set policy-options community com200 members target:1:200
set routing-instances vrf_vlan100 vrf-import import-inet
set routing-instances vrf_vlan100 vrf-export export-inet1
set routing-instances vrf_vlan100 routing-options auto-export family inet unicast
set routing-instances vrf_vlan200 vrf-import import-inet
set routing-instances vrf_vlan200 vrf-export export-inet2
set routing-instances vrf_vlan200 routing-options auto-export family inet unicast
```

Configuring Route Leaking

Step-by-Step Procedure

To configure route leaking on leaf 1:

1. Configure a routing policy that specifies that routes learned through IRB interface irb.100 are exported and then imported into the routing table for vrf_vlan200. Configure another routing policy that specifies that routes learned through IRB interface irb.200 are exported and then imported into the routing table for vrf_vlan100.

   ```
   [edit policy-options]
   user@switch# set policy-statement export-inet1 term 1 from interface irb.100
   user@switch# set policy-statement export-inet1 term 1 then community add com200
   user@switch# set policy-statement export-inet1 term 1 then accept
   user@switch# set policy-statement export-inet2 term 1 from interface irb.200
   user@switch# set policy-statement export-inet2 term 1 then community add com100
   user@switch# set policy-statement export-inet2 term 1 then accept
   user@switch# set policy-statement import-inet term 1 from community com100
   user@switch# set policy-statement import-inet term 1 from community com200
   user@switch# set policy-statement import-inet term 1 then accept
   user@switch# set community com100 members target:1:100
   user@switch# set community com200 members target:1:200
   ```

2. In the VRF routing instances for VLANs v100 and v200. apply the routing policies configured in step 1.

   ```
   [edit routing-instances]
   user@switch# set vrf_vlan100 vrf-import import-inet
   user@switch# set vrf_vlan100 vrf-export export-inet1
   user@switch# set vrf_vlan200 vrf-import import-inet
   user@switch# set vrf_vlan200 vrf-export export-inet2
   ```

3. Specify that unicast routes are to be exported from the vrf_vlan100 routing table into the vrf_vlan200 routing table and vice versa.

   ```
   [edit routing-instances]
   user@switch# set vrf_vlan100 routing-options auto-export family inet unicast
   user@switch# set vrf_vlan200 routing-options auto-export family inet unicast
   ```

Release History Table

Release	Description
17.3R1	Starting with Junos OS Release 17.3R1, the QFX5110 switch can function as a leaf device, which acts as Layer 2 and 3 VXLAN gateways in an EVPN-VXLAN topology with a collapsed IP fabric.

Related Documentation

- Example: Configuring a QFX5110 Switch as a Layer 3 VXLAN Gateway in an EVPN-VXLAN Topology with a Two-Layer IP Fabric on page 353

CHAPTER 13

Configuring a Virtual Chassis and Virtual Chassis Fabric With EVPN-VXLAN

EVPN-VXLAN Support of Virtual Chassis and Virtual Chassis Fabric

A leaf device in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) topology with a two-layer IP fabric can be a standalone switch, or switches configured as members of a Virtual Chassis or a Virtual Chassis Fabric (VCF). Starting with Junos OS Release 14.1X53-D40, a QFX5100 can function as a standalone switch, and multiple QFX5100 switches can function as members of a Virtual Chassis or a Virtual Chassis Fabric (VCF). Starting with Junos OS Release 18.2R1, an EX4600 can function as a standalone switch, and multiple EX4600 switches can function as members of a Virtual Chassis. The Virtual Chassis or VCF function as a single leaf device with Layer 2 VXLAN tunnel endpoint (VTEP) capabilities.

NOTE: You cannot configure EX4600 switches as members of a VCF.

In addition, you cannot configure a mix of QFX5100 and EX4600 switches as members of the same Virtual Chassis.

EVPN supports multihoming active-active mode, which enables a host to be connected to multiple leaf devices through a Layer 2 link aggregation group (LAG) interface. In this situation, the LAG interface can connect a host to multiple standalone leaf devices, Virtual Chassis, VCs, or a mix of these entities.

Regardless of whether a leaf device is a standalone switch or configured as a Virtual Chassis or VCF, on each leaf device, the LAG interface is configured with the same Ethernet segment identifier (ESI) for the host. The leaf devices on which the same ESI is configured are peers to each other.

Implementing multihoming active-active mode in an EVPN-VXLAN topology with Virtual Chassis or VCFs has implications for the sending and receiving of Layer 2 broadcast, unknown unicast, and multicast (BUM) traffic.

Through the control of the LAG interface, a multihomed host, for example, Host 1, sends only one copy of a BUM packet to one of the leaf devices to which it is connected. Conversely, when receiving BUM packets, Host 1 receives only one copy of each packet from one of the leaf devices to which it is connected.

This topic includes the following use cases that describe how multihomed hosts receive Layer 2 BUM packets:

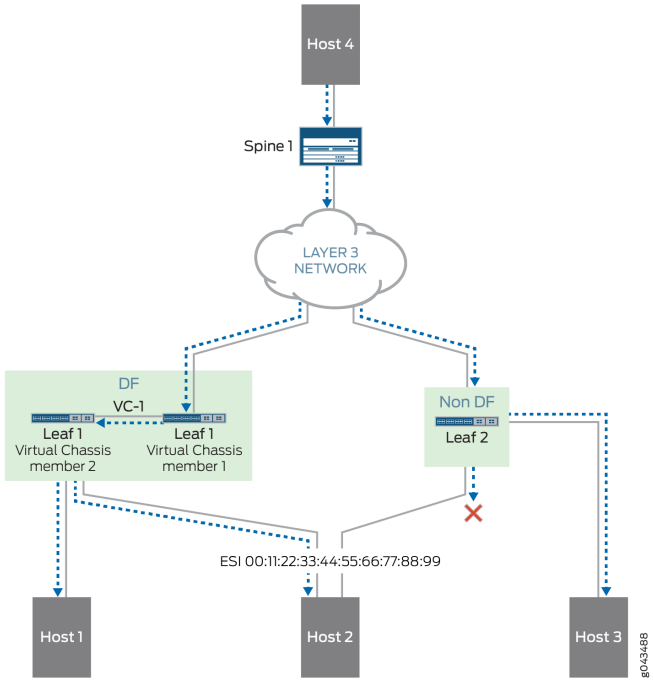
- Use Case 1: Multihomed Host Receives BUM Packets Only from Designated Forwarder (Virtual Chassis) on page 475
- Use Case 2: Multihomed Host Receives BUM Packets Only from Designated Forwarder (VCF) on page 476
- Use Case 3: Local Bias Prevents Multihomed Host from Receiving Same BUM Traffic that It Originates (Virtual Chassis and VCF) on page 479
- Use Case 4: Local Bias Prevents Multihomed Host from Receiving BUM Packets from Both Leaf Device (Virtual Chassis and VCF) on page 481

Use Case 1: Multihomed Host Receives BUM Packets Only from Designated Forwarder (Virtual Chassis)

In the EVPN-VXLAN topology shown in Figure 38 on page 475, there are four hosts connected to spine or leaf devices through a Layer 2 connection. All four hosts are in the same VXLAN. The spine and leaf devices function as virtual tunnel endpoints (VTEPs) that encapsulate and de-encapsulate all Layer 2 packets, including BUM packets, with a VXLAN header.

In this topology, there are two leaf devices. Leaf 1 is a Virtual Chassis that includes switches known as Virtual Chassis members 1 and 2, and Leaf 2 is a standalone switch. In this topology, Host 2 is connected to the two leaf devices through a Layer 2 LAG interface. The ESI for this interface is 00:11:22:33:44:55:66:77:88:99. Per multihoming active-active mode, the two leaf devices are peers, and one of the leaf devices, for example, Leaf 1, is elected as a designated forwarder (DF).

Figure 38: Multihomed Host 2 Receives BUM Packets Only from Leaf 1 (Virtual Chassis)



If Host 4 sends a BUM packet, the following packet flow occurs:

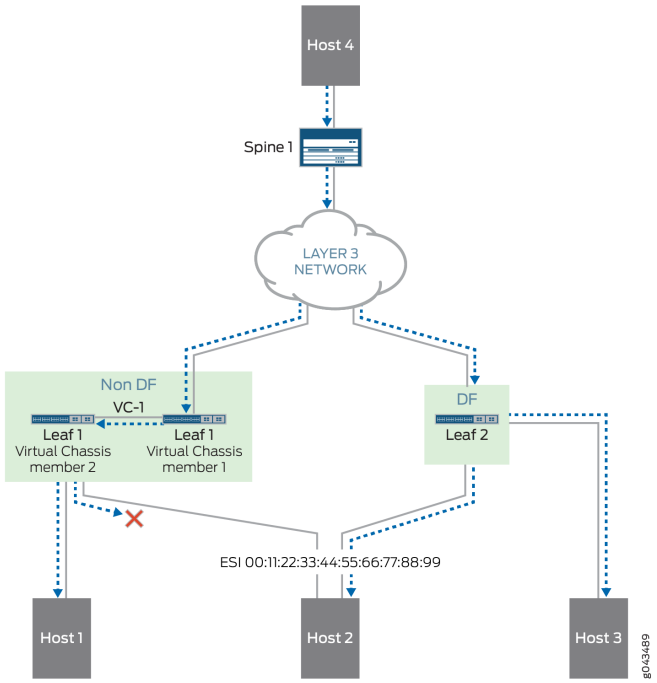
- Spine 1 encapsulates the Layer 2 packet with a VXLAN header and forwards the resulting Layer 3 packet to Leafs 1 and 2.
- Upon receipt of the Layer 3 packet, Leaf 1 de-encapsulates the packet and checks its MAC table to find the destination host. Since the packet is a BUM packet, the packet is to be forwarded to all hosts in the VXLAN. Hosts 1 and 2 are not included in the Virtual Chassis member 1, so it sends the BUM packet through the Virtual Chassis link to Virtual Chassis member 2. Virtual Chassis member 2 forwards the BUM packet to Host 1, and

because Leaf 1 is the DF for ESI 00:11:22:33:44:55:66:77:88:99, Virtual Chassis member 2 forwards the packet to Host 2.

- Upon receipt of the Layer 3 packet, Leaf 2 de-encapsulates the packet and checks its MAC table to find the destination host. Since the packet is a BUM packet, the packet is to be forwarded to all hosts in the VXLAN. Therefore, Leaf 2 forwards the BUM packet to Host 3, and because Leaf 2 is not the DF for ESI 00:11:22:33:44:55:66:77:88:99, Leaf 2 drops the packet intended for Host 2.

Figure 39 on page 476 shows the same EVPN-VXLAN topology as in Figure 38 on page 475 except that in Figure 39 on page 476, Leaf 2 is the DF for ESI 00:11:22:33:44:55:66:77:88:99.

Figure 39: Multihomed Host 2 Receives BUM Packets Only from, Leaf 2 (Virtual Chassis)



Therefore, the packet flow for the topology in Figure 39 on page 476 is the same as that in Figure 38 on page 475 except for the following:

- Because Leaf 1 is not the DF for ESI 00:11:22:33:44:55:66:77:88:99, Leaf 1 drops the packet intended for Host 2.
- Because Leaf 2 is the DF for ESI 00:11:22:33:44:55:66:77:88:99, Leaf 2 forwards the packet to Host 2.

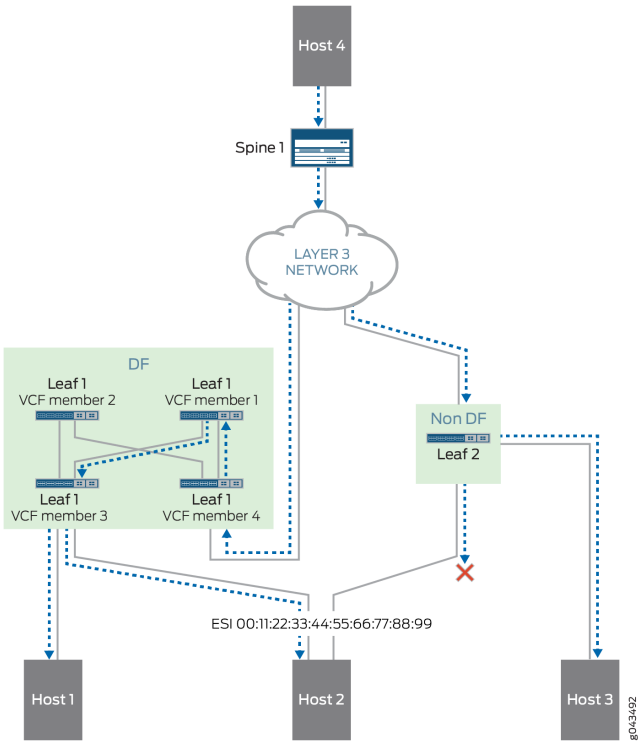
Use Case 2: Multihomed Host Receives BUM Packets Only from Designated Forwarder (VCF)

NOTE: You cannot configure EX4600 switches as members of a VCF.

In the EVPN-VXLAN topology shown in Figure 40 on page 477, there are four hosts connected to spine or leaf devices through a Layer 2 connection. All four hosts are in the same VXLAN. The spine and leaf devices function as VTEPs that encapsulate and de-encapsulate all Layer 2 packets, including BUM packets, with a VXLAN header.

In this topology, there are two leaf devices. Leaf 1 is a VCF that includes switches known as VCF members 1 though 4; and Leaf 2 is a standalone switch. In this topology, Host 2 is connected to the two leaf devices through a Layer 2 LAG interface. The ESI for this interface is 00:11:22:33:44:55:66:77:88:99. Per multihoming active-active mode, the two leaf devices are peers, and one of the leaf devices, for example, Leaf 1, is elected as a DF.

Figure 40: Multihomed Host 2 Receives BUM Packets Only from Leaf 1 (VCF)



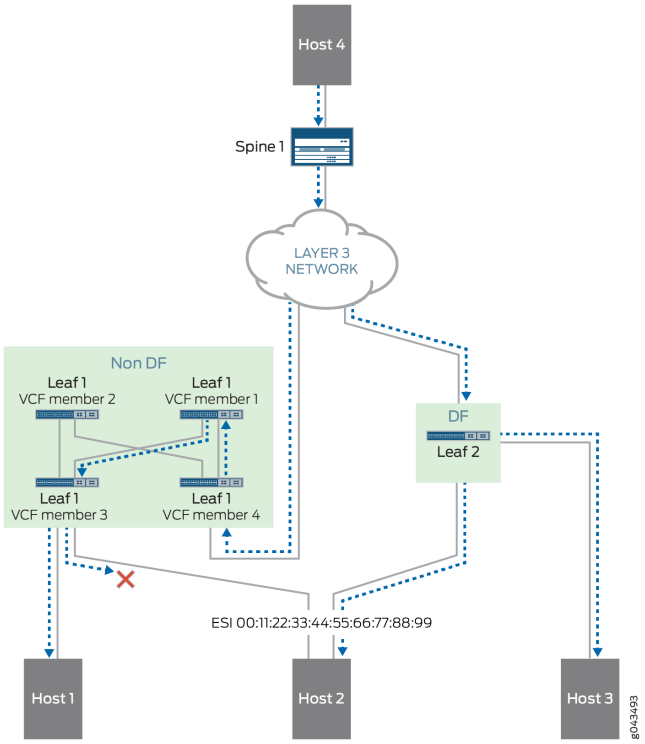
If Host 4 sends a BUM packet, the following packet flow occurs:

- Spine 1 encapsulates the Layer 2 packet with a VXLAN header and forwards the resulting Layer 3 packet to Leafs 1 and 2.
- Upon receipt of the Layer 3 packet, Leaf 1 de-encapsulates the packet and checks its MAC table to find the destination host. Since the packet is a BUM packet, the packet is to be forwarded to all hosts in the VXLAN. Hosts 1 and 2 are not included in VCF member 4, and although VCF member 4 can forward the BUM packet to either VCF member 1 or 2, in this case, it sends the packet through the VCF link to VCF member 1. VCF member 1 forwards the BUM packet to VCF member 3, which in turn forwards the packet to Host 1. Because Leaf 1 is the DF for ESI 00:11:22:33:44:55:66:77:88:99, Leaf 1 forwards the packet to Host 2.

- Upon receipt of the Layer 3 packet, Leaf 2 de-encapsulates the packet and checks its MAC table to find the destination host. Since the packet is a BUM packet, the packet is to be forwarded to all hosts in the VXLAN. Therefore, Leaf 2 forwards the BUM packet to Host 3, and because Leaf 2 is not the DF for ESI 00:11:22:33:44:55:66:77:88:99, Leaf 2 drops the packet intended for Host 2.

Figure 41 on page 478 shows the same EVPN-VXLAN topology as in Figure 40 on page 477 except that in Figure 41 on page 478, Leaf 2 is the DF for ESI 00:11:22:33:44:55:66:77:88:99.

Figure 41: Multihomed Host 2 Receives BUM Packets Only from Leaf 2 (Virtual Chassis Fabric)



The packet flow for the topology in Figure 41 on page 478 is the same as that in Figure 40 on page 477 except for the following:

- Because Leaf 1 is not the DF for ESI 00:11:22:33:44:55:66:77:88:99, Leaf 1 drops the packet intended for Host 2.
- Because Leaf 2 is the DF for ESI 00:11:22:33:44:55:66:77:88:99, Leaf 2 forwards the packet to Host 2.

Use Case 3: Local Bias Prevents Multihomed Host from Receiving Same BUM Traffic that It Originates (Virtual Chassis and VCF)

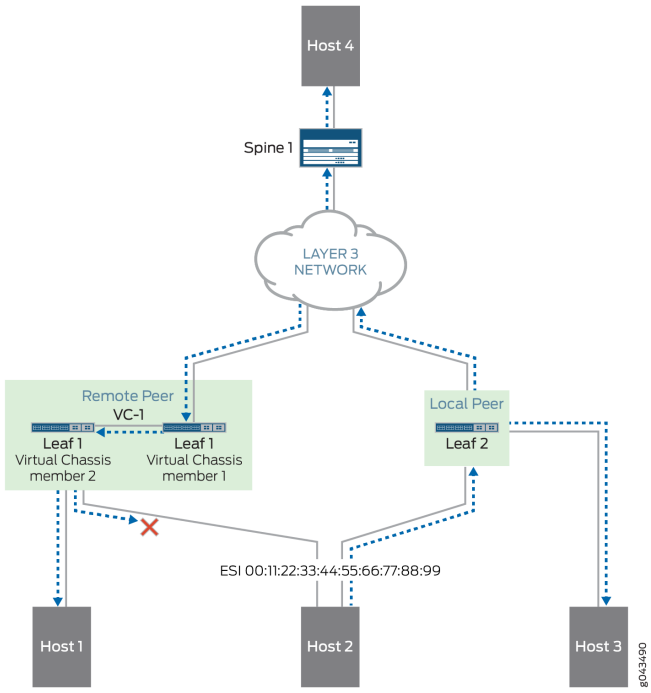
NOTE: You cannot configure EX4600 switches as members of a VCF.

In addition, you cannot configure a mix of QFX5100 and EX4600 switches as members of the same Virtual Chassis.

In the EVPN-VXLAN topology shown in Figure 42 on page 479, there are four hosts connected to spine or leaf devices through a Layer 2 connection. All four hosts are in the same VXLAN. The spine and leaf devices function as VTEPs that encapsulate and de-encapsulate all Layer 2 packets, including BUM packets, with a VXLAN header.

In this topology, there are two leaf devices. Leaf 1 is a Virtual Chassis that includes switches known as Virtual Chassis members 1 and 2, and Leaf 2 is a standalone switch. In this topology, Host 2 is connected to the two leaf devices through a Layer 2 LAG interface. The ESI for this interface is 00:11:22:33:44:55:66:77:88:99. Per multihoming active-active mode, the two leaf devices are peers.

Figure 42: Local Bias Prevents Multihomed Host 2 from Receiving Same BUM Traffic that It Originates (Virtual Chassis)



In this use case, Host 2 sends a BUM packet that must be forwarded to all hosts in the VXLAN. To prevent multihomed Host 2 from receiving the same BUM packet that it originated, there is a local bias, which determines that the local peer forwards the BUM

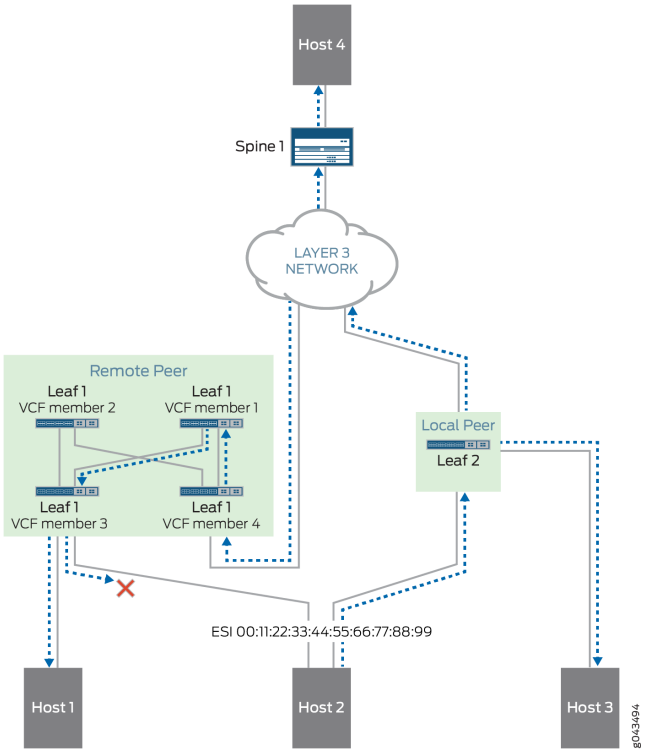
packet and the remote peer drops it. This behavior of the local peer forwarding the packet occurs regardless of the DF status (DF or non-DF) of the local peer.

To illustrate the local bias, if Host 2 sends a BUM packet, the following packet flow occurs:

- Upon receipt of the Layer 2 BUM packet, Leaf 2 does the following:
  - Forwards the packet to Host 3.
  - Encapsulates the Layer 2 packet with a VXLAN header and forwards the resulting Layer 3 packet to Spine 1 and Leaf 1.
- Upon receipt of the Layer 3 packet, Leaf 1 de-encapsulates the packet and checks its MAC table to find the destination host. Since the packet is a BUM packet, the packet is to be forwarded to all hosts in the VXLAN. Hosts 1 and 2 are not included in Virtual Chassis member 1, so Virtual Chassis member 1 sends the BUM packet through the Virtual Chassis link to Virtual Chassis member 2. which in turn forwards the BUM packet to Host 1. However, since Leaf 1 is the remote peer to Leaf 2, which received the original BUM packet from Host 2, Leaf 1 drops the packet intended for Host 2.

Figure 43 on page 480 shows the same EVPN-VXLAN topology as in Figure 42 on page 479 except that in Figure 43 on page 480, Leaf 1 is configured as a VCF that includes switches that are known as VCF members 1 through 4.

Figure 43: Local Bias Prevents Multihomed Host 2 from Receiving Same BUM Traffic that It Originates (VCF)



As a result, the packet flow for the topology in Figure 43 on page 480 is the same as that for topology in Figure 42 on page 479 except for the following:

- Upon receipt of the Layer 3 packet, Leaf 1 de-encapsulates the packet and checks its MAC table to find the destination host. Hosts 1 and 2 are not included in VCF member 4, and although VCF member 4 can send the BUM packet to VCF member 1 or 2, in this case, it sends the packet to VCF member 1. VCF member 1 forwards the BUM packet to VCF member 3, which in turn forwards the packet to Host 1. However, since Leaf 1 is the remote peer to Leaf 2, which received the original BUM packet from Host 2, Leaf 1 drops the packet intended for Host 2.

Use Case 4: Local Bias Prevents Multihomed Host from Receiving BUM Packets from Both Leaf Device (Virtual Chassis and VCF)

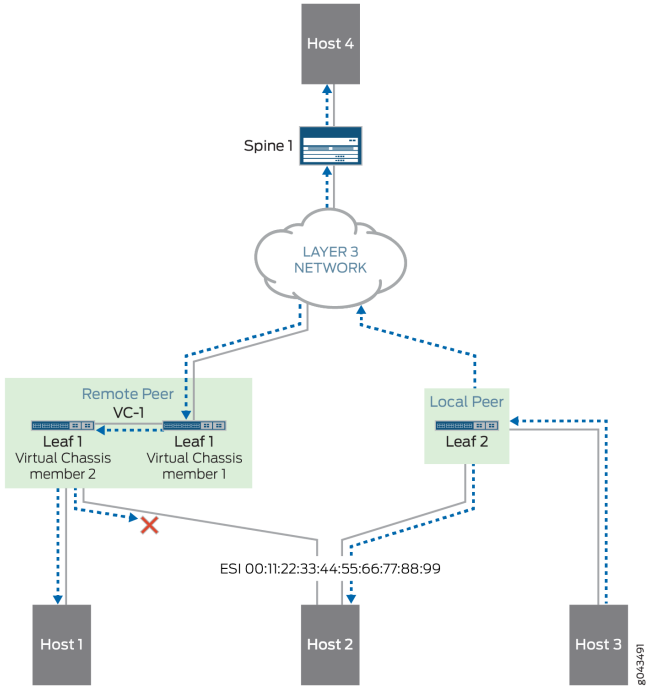
NOTE: You cannot configure EX4600 switches as members of a VCF.

In addition, you cannot configure a mix of QFX5100 and EX4600 switches as members of the same Virtual Chassis.

In the EVPN-VXLAN topology shown in Figure 44 on page 482, there are four hosts connected to spine or leaf devices through a Layer 2 connection. All four hosts are in the same VXLAN. The spine and leaf devices function as VTEPs that encapsulate and de-encapsulate all Layer 2 packets, including BUM packets, with a VXLAN header.

In this topology, there are two leaf devices. Leaf 1 is a Virtual Chassis that includes switches known as Virtual Chassis members 1 and 2, and Leaf 2 is a standalone switch. In this topology, Host 2 is connected to the two leaf devices through a Layer 2 LAG interface. The ESI for this interface is 00:11:22:33:44:55:66:77:88:99. Per multihoming active-active mode, the two leaf devices are peers.

Figure 44: Local Bias Prevents Multihomed Host 2 from Receiving BUM Packets from Both Leaf Devices (Virtual Chassis)



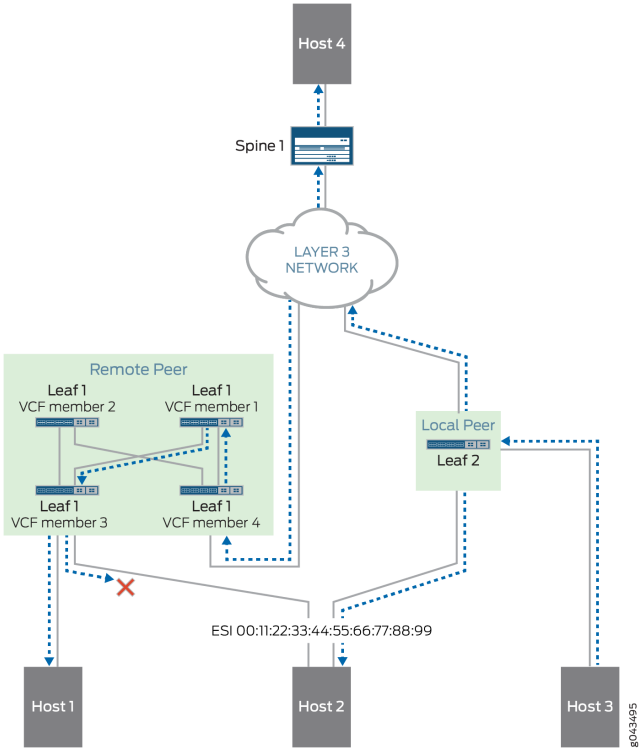
In this use case, Host 3 sends a BUM packet that must be forwarded to all hosts in the VXLAN. To prevent multihomed Host 2 from receiving this BUM packet from both Leafs 1 and 2, there is a local bias, which determines that the local peer forwards the BUM packet and the remote peer drops it. This behavior of the local peer forwarding the packet occurs regardless of the DF status (DF or non-DF) of the local peer.

To illustrate the local bias, if Host 3 sends a BUM packet, the following packet flow occurs:

- Upon receipt of the BUM packet, Leaf 2 does the following:
  - Forwards the packet to Host 2. Since Host 3 is connected to the same leaf device (Leaf 2) as multihomed Host 2, Leaf 2 is considered the local peer, and according to the local bias, the local peer takes precedence over the remote peer, which in this case is Leaf 1.
  - Encapsulates the Layer 2 packet with a VXLAN header and forwards the resulting Layer 3 packet to Spine 1 and Leaf 1.
- Upon receipt of the Layer 3 packet, Leaf 1 de-encapsulates the packet and checks its MAC table to find the destination host. Hosts 1 and 2 are not included in Virtual Chassis member 1, so Virtual Chassis member 1 sends the BUM packet through the Virtual Chassis link to Virtual Chassis member 2. which in turn forwards the BUM packet to Host 1. However, when it comes to forwarding the packet to Host 2, Leaf 1, which is considered to be the remote peer, drops the packet because the packet is forwarded by the local peer, which is Leaf 2.

Figure 45 on page 483 shows the same EVPN-VXLAN topology as in Figure 44 on page 482 except that in Figure 45 on page 483, Leaf 1 is configured as a VCF that includes switches that are known as VCF members 1 through 4.

Figure 45: Local Bias Prevents Multihomed Host 2 from Receiving BUM Packets from Both Leaf Devices (VCF)



As a result, the packet flow for the topology in Figure 45 on page 483 is the same as that for the topology in Figure 44 on page 482 except for the following:

- Upon receipt of the Layer 3 packet, Leaf 1 de-encapsulates the packet and checks its MAC table to find the destination host. Hosts 1 and 2 are not included in VCF member 4, and although VCF member 4 can send the BUM packet to either VCF members 1 or 2, in this case, VCF member 4 forwards the BUM packet to VCF member 1. VCF member 4 forwards the packet to VCF member 3, which in turn forwards the packet to Host 1. However, when it comes to forwarding the packet to Host 2, Leaf 1, which is considered to be the remote peer, drops the packet because the packet is forwarded by the local peer, which is Leaf 2.

Release History Table

Release	Description
18.2R1	Starting with Junos OS Release 18.2R1, an EX4600 can function as a standalone switch, and multiple EX4600 switches can function as members of a Virtual Chassis.
14.1X53-D40	Starting with Junos OS Release 14.1X53-D40, a QFX5100 can function as a standalone switch, and multiple QFX5100 switches can function as members of a Virtual Chassis or a Virtual Chassis Fabric (VCF).

CHAPTER 14

Configuring Multicast Features

Overview of IGMP Snooping in an EVPN-VXLAN Environment

IGMP snooping is used to constrain multicast traffic in a broadcast domain to interested receivers and multicast devices. In an environment with significant multicast traffic, using IGMP snooping preserves bandwidth because multicast traffic is forwarded only on those interfaces where there are IGMP listeners. Starting with Junos OS Release 17.2R1, IGMP snooping is supported in an Ethernet VPN (EVPN) environment on QFX10000 switches. Previously, IGMP snooping was supported only with virtual LANs. As a result, you can now configure IGMP snooping in an EVPN-Virtual Extensible LAN (VXLAN) environment.

IGMP snooping in an EVPN-VXLAN multihomed environment is useful in data center interconnect (DCI) scenarios with a high level of multicast traffic. Every customer edge (CE) device does not need to receive every instance of multicast traffic. Enabling a provider edge (PE) device to send multicast traffic to a CE device only as needed utilizes the links between CE and PE devices more efficiently. When multicast traffic arrives at the VXLAN core, a remote PE device configured with EVPN forwards traffic only to the access interfaces where there are IGMP listeners. The multicast senders and receivers can be on the same site or on different sites. A site can have either only receiver, only source or both source and receiver attached to it. This implementation relies on IRB interfaces. Both intra-VLAN and inter-VLAN forwarding are supported. Use Protocol Independent Multicast (PIM) for inter-VLAN forwarding.

In this implementation, every link between a CE and a PE device must support IGMP snooping. Only proxy mode is supported with IGMP snooping. All multihomed interfaces must have the same configuration. Only the active-active mode of multihoming is supported. Active-standby mode is not supported. Only IGMP version 2 is supported. IGMP versions 1 and 3 are not supported. Additionally, the **multicast-options** configuration stanza is supported.

Starting with Junos OS Release 17.3R1, you can configure the PE device to perform inter-VLAN forwarding of multicast traffic without having to configure IRB interfaces. In such a scenario, an external multicast router is used to send IGMP queries to solicit reports

and to forward VLAN traffic through a Layer 3 multicast protocol such as PIM. IRB interfaces are not supported with the use of an external multicast router.

Also starting with Junos OS Release 17.3R1, you can connect a PE device to an external Layer 3 device running PIM. This implementation relies on IRB interfaces and supports sending and receiving multicast traffic to and from the data center through an external gateway running PIM. To enable the PEs to forward traffic to the external domain, configure PIM-to-IGMP translation by including the **pim-to-igmp-proxy upstream-interface *irb-interface-name*** statements at the **[edit routing-options multicast]** hierarchy level. Additionally, you an now configure PIM on the PE so that it functions only to forward inter-VLAN traffic within the data center. This means that you do not need to configure a PIM rendezvous point since forming PIM adjacencies is not required. The gateway device only needs to view the data center as a Layer 2 multicast domain. Include the **passive** statement at the **[edit protocols pim]** hierarchy level.

Release History Table

Release	Description
17.3R1	Starting with Junos OS Release 17.3R1, you can configure the PE device to perform inter-VLAN forwarding of multicast traffic without having to configure IRB interfaces.
17.3R1	Also starting with Junos OS Release 17.3R1, you can connect a PE device to an external Layer 3 device running PIM.
17.2R1	Starting with Junos OS Release 17.2R1, IGMP snooping is supported in an Ethernet VPN (EVPN) environment on QFX10000 switches.

Related Documentation

- *distributed-dr*
- *igmp-snooping*
- *multicast-router-interface*
- Example: Preserving Bandwidth with IGMP Snooping in an EVPN-VXLAN Environment on page 486

Example: Preserving Bandwidth with IGMP Snooping in an EVPN-VXLAN Environment

This example shows how to configure IGMP snooping on provider edge (PE) devices in an Ethernet VPN (EVPN)-Virtual Extensible LAN. When multicast traffic arrives at the VXLAN core, a PE device configured with EVPN forwards traffic only to the local access interfaces where there are IGMP listeners.

Requirements

This example uses the following hardware and software components:

- Two QFX10000 switches configured as multihomed PE devices that are connected to the CE, one QFX10000 device configured as a PE device connected to the multihomed PEs and a QFX5110 configured as a CE device.
- Junos OS Release 17.2R1 or later running on all devices.

Overview

IGMP snooping is used to constrain multicast traffic in a broadcast domain to interested receivers and multicast devices. In an environment with significant multicast traffic, IGMP snooping preserves bandwidth because multicast traffic is forwarded only on those interfaces where there are IGMP listeners.

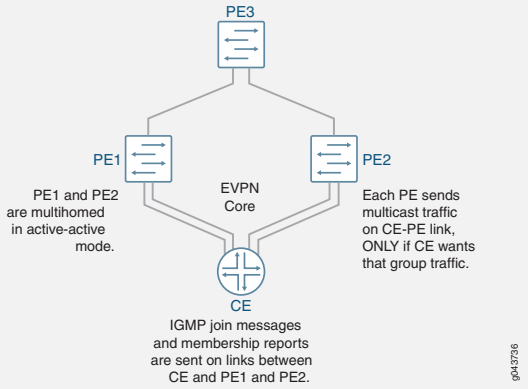
Multicast traffic is flooded in the EVPN core even if there are no remote receivers. For multihomed peer PE devices, only active-active mode is supported. IGMP snooping in enabled on each VLAN, which automatically enables Type-7 and Type-8 network layer reachability information (NLRI) to sync IGMP joins and leaves. IGMP is enabled to manage multicast group membership. Only IGMP version 2 is supported. Versions 1 and 3 are not supported. You must also configure IGMP snooping in proxy mode for the PE to become the IGMP querier for the local access interfaces.

This feature supports both intra-VLAN and inter-VLAN forwarding. You can configure the PE device to perform either or both. In this example, to enable inter-VLAN forwarding, each PE device is configured as a statically defined Protocol Independent Multicast (PIM) rendezvous point (RP) to enable multicast forwarding. You also configure the **distributed-dr** statement at the **[edit protocols pim interface *interface-name*]** hierarchy level for each IRB interface. This statement enables PIM by forward multicast traffic more efficiently by disabling PIM features that are not required in this scenario. When you configure this statement, PIM ignores the designated router (DR) status of the interface when processing IGMP reports received on the interface. When the interface receives the IGMP report, the PE device sends PIM upstream join messages to pull the multicast stream and forward it to the interface—regardless of the DR status of the interface.

Topology

Figure 46 on page 488 illustrates an EVPN-VXLAN environment where two PE devices (PE1 and PE2) are connected to the customer edge (CE) device. These PEs are multihomed in active-active mode to provide redundancy. A third PE device forwards traffic to the PE devices that face the CE. IGMP is enabled on integrated routing and bridging (IRB) interfaces. IGMP snooping is enabled on the VLANs. Because this implementation does not support the use of a multicast router, each VLAN in the PE is enabled as an IGMP Layer 2 querier. The multihomed PE devices forward traffic towards the CE only on those interfaces where there are IGMP listeners. The CE device hosts five VLANs.

Figure 46: IGMP Snooping in an EVPN-VXLAN Environment



Configuration

To configure IGMP Snooping in an EVPN-VXLAN environment, perform these tasks:

- Configuring PE1 on page 494
- Configuring PE2 on page 502
- Configuring CE Device on page 511
- Configuring PE3 on page 514

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

Device PE1

```
set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/0:1 description “Connected to CE”
set interfaces xe-0/0/0:2 ether-options 802,3ad ae1
set interfaces xe-0/0/1:0 enable
set interfaces xe-0/0/1:0 unit 0 description “Connected to PE3”
set interfaces xe-0/0/1:0 unit 0 family inet address 192.0.2.1/24
set interfaces ae0 enable
set interfaces ae1 enable
set interfaces lo0 unit 0 family inet address 192.168.1.1/32
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 0 aggregated ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 0 description “Connected to CE”
set interfaces ae1 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ae1 esi all-active
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options periodic fast
set interfaces ae1 aggregated-ether-options lacp system id 00:00:00:00:00:02
set interfaces ae1 unit 0 description “Connected to CE”
```

```
set interfaces ae0 family ethernet-switching interface-mode trunk
set interfaces ae0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3 VLAN4
  VLAN5 ]
set interfaces ae1 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 0 family ethernet-switching vlan [ VLAN1 VLAN2 VLAN3 VLAN4
  VLAN5 ]
set interfaces irb unit 1 family inet address 10.1.1.1/24 virtual-gateway-address 10.1.1.10
set interfaces irb unit 2 family inet address 10.2.1.1/24 virtual-gateway-address 10.2.1.10
set interfaces irb unit 3 family inet address 10.3.1.1/24 virtual-gateway-address 10.3.1.10
set interfaces irb unit 4 family inet address 10.4.1.1/24 virtual-gateway-address 10.4.1.10
set interfaces irb unit 5 family inet address 10.5.1.1/24 virtual-gateway-address 10.5.1.10
set routing-options router-id 192.168.1.1
set routing-options autonomous-system 65536
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols bgp group INT type internal
set protocols bgp group INT local-address 192.168.1.1
set protocols bgp group INT family evpn signaling
set protocols bgp group INT local-as 65536
set protocols bgp group INT neighbor 172.16.1.1
set vlans VLAN vlan-id 1
set vlans VLAN1 l3-interface irb.1
set vlans VLAN1 vxlan vni 1
set vlans VLAN1 vxlan ingress-node-replication
set vlans VLAN2 vlan-id 2
set vlans VLAN2 l3-interface irb.2
set vlans VLAN2 vxlan vni2
set vlans VLAN2 vxlan ingress-node-replication
set vlans VLAN3 vlan-id 3
set vlans VLAN3 l3-interface irb.3
set vlans VLAN3 vxlan vni 3
set vlans VLAN3 vxlan ingress-node-replication
set vlans VLAN4 vlan-id 4
set vlans VLAN4 l3-interface irb.4
set vlans VLAN4 vxlan vni 4
set vlans VLAN4 vxlan ingress-node-replication
set vlans VLAN5 vlan-id 5
set vlans VLAN5 l3-interface irb.5
set vlans VLAN5 vxlan vni 5
set vlans VLAN5 vxlan ingress-node-replication
set protocols evpn encapsulation vxlan
set protocols evpn ingress-replication
set protocols evpn extended-vini-list 1-5
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-options forwarding-table export evpn-pplb
set switch-options vtep-source-interface lo0:0
set switch-options route distinguisher 192.168.1.1:1
set switch-options vrf-target target:1:1
set protocols igmp interface irb.1
set protocols igmp interface irb.2
set protocols igmp interface irb.3
set protocols igmp interface irb.4
set protocols igmp interface irb.5
set protocols igmp-snooping vlan VLAN1 l2-querier source-address 10.1.1.1
```

```
set protocols igmp-snooping vlan VLAN1 proxy
set protocols igmp-snooping vlan VLAN2 l2-querier source-address 10.2.1.1
set protocols igmp-snooping vlan VLAN2 proxy
set protocols igmp-snooping vlan VLAN3 l2-querier source-address 10.3.1.1
set protocols igmp-snooping vlan VLAN3 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.4.1.1.
set protocols igmp-snooping vlan VLAN4 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.5.1.1.
set protocols igmp-snooping vlan VLAN5 proxy
set protocols pim rp static address 172.16.1.1
set protocols pim interface irb.1 distributed-dr
set protocols pim interface irb.2 distributed-dr
set protocols pim interface irb.3 distributed-dr
set protocols pim interface irb.4 distributed-dr
set protocols pim interface irb.5 distributed-dr
```

Device PE2

```
set chassis aggregated-devices ethernet device-count 2
set interfaces xe-0/0/0:0 ether-options 802.3ad ae1
set interfaces xe-0/0/0:1 description “Connected to CE”
set interfaces xe-0/0/0:1 ether-options 802.3ad ae0
set interfaces xe-0/0/1:0 enable
set interfaces xe-0/0/1:0 unit 0 description “Connected to PE3”
set interfaces xe-0/0/1:0 family inet address 198.51.100.1/24
set interfaces ae0 enable
set interfaces ae1 enable
set interfaces lo0 unit 0 family inet address 192.168.2.1/32
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
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 00:00:00:00:00:01
set interfaces ae1 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ae1 esi all-active
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces ae1 aggregated-ether-options lacp system-id 00:00:00:00:00:02
set interfaces ae0 unit 0 description “Connected to CE”
set interfaces ae0 unit family ethernet-switching interface-mode trunk
set interfaces ae0 unit family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3
  VLAN4 VLAN5 ]
set interfaces ae1 unit 0 description “Connected to CE”
set interfaces ae1 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2 VLAN3
  VLAN4 VLAN5 ]
set interfaces irb unit 1 family inet address 10.1.1.2/24 virtual-gateway-address 10.1.1.10
set interfaces irb unit 2 family inet address 10.2.1.2/24 virtual-gateway-address 10.2.1.10
set interfaces irb unit 3 family inet address 10.3.1.2/24 virtual-gateway-address 10.3.1.10
set interfaces irb unit 4 family inet address 10.4.1.2/24 virtual-gateway-address 10.4.1.10
set interfaces irb unit 5 family inet address 10.5.1.2/24 virtual-gateway-address 10.5.1.10
set routing-options router-id 192.168.2.1
set routing-options autonomous-system 65536
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/1:0.0
```

```
set protocols bgp group INT type internal
set protocols bgp group INT local-address 192.168.2.1
set protocols bgp group INT family evpn signaling
set protocols bgp group INT local-as 65536
set protocols bgp group INT neighbor 172.16.1.1
set vlans VLAN1 vlan-id 1
set vlans VLAN1 l3-interface irb.1
set vlans VLAN1 vxlan vni 1
set vlans VLAN1 vxlan ingress-node-replication
set vlans VLAN2 vlan-id 2
set vlans VLAN2 l3-interface irb.2
set vlans VLAN2 vxlan vni 2
set vlans VLAN2 vxlan ingress-node-replication
set vlans VLAN3 vlan-id 3
set vlans VLAN3 l3-interface irb.3
set vlans VLAN3 vxlan vni 3
set vlans VLAN3 vlxan ingress-node-replication
set vlans VLAN4 vlan-id 4
set vlans VLAN4 l3-interface irb.4
set vlans VLAN4 vxlan vni 4
set vlans VLAN4 vlxan ingress-node-replication
set vlans VLAN5 vlan-id 5
set vlans VLAN5 l3-interface irb.5
set vlans VLAN5 vxlan vni 3
set vlans VLAN5 vlxan ingress-node-replication
set protocols evpn encapsulation vxlan
set protocols evpn muticast-mode ingress-encapsulation
set protocols evpn extended-vni-list 1-5
set policy-options policy statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-options forwarding-table export evpn-pplb
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 192.168.2.1:1
set switch-options vrf-target target:1:1
set protocols igmp interface irb.1
set protocols igmp interface irb.2
set protocols igmp interface irb.3
set protocols igmp interface irb.4
set protocols igmp interface irb.5
set protocols igmp-snooping vlan VLAN1 l2-querier source-address 10.1.1.2
set protocols igmp-snooping vlan VLAN1 proxy
set protocols igmp-snooping vlan VLAN2 l2-querier source-address 10.2.1.2
set protocols igmp-snooping vlan VLAN2 proxy
set protocols igmp-snooping vlan VLAN3 l2-querier source-address 10.3.1.2
set protocols igmp-snooping vlan VLAN3 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.4.1.2
set protocols igmp-snooping vlan VLAN4 proxy
set protocols igmp-snooping vlan VLAN5 l2-querier source-address 10.5.1.2
set protocols igmp-snooping vlan VLAN5 proxy
set protocols pim rp static address 172.16.1.1
set protocols pim interface irb.1 distributed-dr
set protocols pim interface irb.2 distributed-dr
set protocols pim interface irb.3 distributed-dr
set protocols pim interface irb.4 distributed-dr
set protocols pim interface irb.5 distributed-dr
```

CE

```
set chassis aggregated devices ethernet device-count 2
set interfaces xe-0/2/0 ether-options 802.3ad ae0
set interfaces xe-0/2/1 ether-options 802.3ad ae0
set interfaces xe-0/2/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/2/0 unit 0 family ethernet-switching vlan members 1-5
set interfaces xe-0/2/1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/2/1 unit 0 family ethernet-switching vlan members 1-5
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 1-5
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces ae1 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 0 family ethernet-switching vlan members 1-5
set vlans BD-1 domain-type bridge
set vlans BD-1 vlan-id 1
set vlans BD-2 domain-type bridge
set vlans BD-2 vlan-id 2
set vlans BD-3 domain-type bridge
set vlans BD-3 vlan-id 3
set vlans BD-4 domain-type bridge
set vlans BD-4 vlan-id 4
set vlans BD-5 domain-type bridge
set vlans BD-5 vlan-id 5
```

PE3

```
set interfaces xe-0/0/0 enable
set interfaces xe-/0/0/0 unit 0 description “Connected to PE1”
set interfaces xe-0/0/0 unit 0 family inet 192.0.2.2/24
set interfaces xe-0/0/1 enable
set interfaces xe-0/0/1 unit 0 description “Connected to PE2”
set interfaces xe-0/0/1 unit 0 family inet 198.51.100.2/24
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set interfaces xe-0/0/0:1 enable
set interfaces xe-0/0/0:1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/0:1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2
  VLAN3 VLAN4 VLAN5 ]
set interfaces xe-0/0/1:1 enable
set interfaces xe-0/0/1:1 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/1:1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2
  VLAN3 VLAN4 VLAN5 ]
set interfaces irb unit 1 famly inet address 10.1.1.5/24 virtual-gateway address 10.1.1.10
set interfaces irb unit 2 family inet address 10.2.1.5/24 virtual-gateway address 10.2.1.10
set interfaces irb unit 3 family inet address 10.3.1.5/24 virtual-gateway address 10.3.1.10
set interfaces irb unit 4 family inet address 10.4.1.5/24 virtual-gateway address 10.4.1.10
set interrfaces irb unit 5 family inet address 10.5.1.5/24 virtual-gateway address 10.5.1.10
set routing-options router-id 172.16.1.1
se routing-options autonomous-system 65536
set ospf area 0.0.0.0 interface all
set ospf area 0.0.0.0 fxp0.0 disable
set protocols bgp group INT type internal
set protocols bgp group INT local-address 172.16.1.1
set protocols bgp group INT family evpn signaling
```

```
set protocols bgp group INT local-as 65536
set protocols bgp group INT neighbor 192.168.1.1
set protocols bgp group INT neighbor 192.168.2.1
set vlans VLAN1 vlan-id 1
set vlans VLAN1 l3-interface irb.1
set vlans VLAN1 vxlan vni 1
set vlans VLAN1 vxlan ingress-node-replication
set vlans VLAN2 vlan-id 2
set vlans VLAN2 l3-interface irb.2
set vlans VLAN2 vxlan vni 2
set vlans VLAN2 vxlan ingress-node-replication
set vlans VLAN3 vlan-id 3
set vlans VLAN3 l3-interface irb.3
set vlans VLAN3 vxlan vni 3
set vlans VLAN3 vxlan ingress-node-replication
set vlans VLAN4 vlan-id 4
set vlans VLAN4 l3-interface irb.4
set vlans VLAN4 vxlan vni 4
set vlans VLAN5 vlxan ingress-node-replication
set vlans VLAN5 vlan-id 5
set vlans VLAN5 l3-interface irb.5
set vlans VLAN5 vxlan vni 5
set vlans VLAN5 vxlan ingress-node-replication
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn extended-vni-list 1-5
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pbllb then load-balance per packet
set routing-options forwarding-table export evpn-pplb
set switch-options vtep-source-interface lo0
set switch-options route-distinguisher 172.16.1.1:1
set switch-option vrf-target target:1:1
set protocols igmp interface irb.1
set protocols igmp interface irb.2
set protocols igmp interface irb.3
set protocols igmp interface irb.4
set protocols igmp interface irb.5
set protocols igmp-snooping vlan VLAN1 l2-querier source-address 10.1.1.5
set protocols igmp-snooping vlan VLAN1 proxy
set protocols igmp-snooping vlan VLAN2 l2-querier source-address 10.2.1.5
set protocols igmp-snooping vlan VLAN2 proxy
set protocols igmp-snooping vlan VLAN3 l2-querier source-address 10.3.1.5
set protocols igmp-snooping vlan VLAN3 proxy
set protocols igmp-snooping vlan VLAN4 l2-querier source-address 10.4.1.5
set protocols igmp-snooping vlan VLAN4 proxy
set protocols igmp-snooping vlan VLAN5 l2-querier source-address 10.5.1.5
set protocols igmp-snooping vlan VLAN5 proxy
set protocols pim rp local 172.16.1.1
set protocols pim interface irb.1 distributed-dr
set protocols pim interface irb.2 distributed-dr
set protocols pim interface irb.3 distributed-dr
set protocols pim interface irb.4 distributed-dr
set protocols pim interface irb.5 distributed-dr
```

Configuring PE1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device PE1:

1. Specify the number of aggregated Ethernet logical interfaces.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 2
```

2. Configure the interfaces.

```
[edit interfaces]
user@PE1# set xe-0/0/0:1 description “Connected to CE”
user@PE1# set xe-0/0/0:2 ether-options 802.3ad ae1
user@PE1# set xe-0/0/1:0 enable
user@PE1# set xe-0/0/1:0 unit 0 description “Connected to PE3”
user@PE1# set xe-0/0/1:0 unit 0 family inet address 192.0.2.1/24
user@PE1# set ae0 enable
user@PE1# set ae1 enable
user@PE2# set lo0 unit 0 family inet address 192.168.1.1/32
```

3. Configure active-active multihoming and enable the Link Aggregation Control Protocol (LACP) on each aggregated Ethernet interface.

```
[edit interfaces]
user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11:11
user@PE1# set esi all-active
user@PE1# set ae0 aggregated-ether-options lacp active
user@PE1# set ae0 aggregated-ether-options lacp periodic-fast
user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
user@PE1# set ae1 esi 00:22:22:22:22:22:22:22:22:22
user@PE1# set ae1 esi all-active
user@PE1# set ae1 aggregated-ether-options lacp active
user@PE1# set ae1 aggregated-ether-options lacp periodic-fast
user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:02
```

4. Configure each aggregated Ethernet interface as a trunk port.

```
[edit interfaces]
user@PE1# set ae0 unit 0 description “Connected to CE”
user@PE1# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@PE1# set ae0 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2
   VLAN3 VLAN4 VLAN5 ]
user@PE1# set ae1 unit 0 description “Connected to CE”
user@PE1# set ae1 unit 0 family ethernet-switching interface-mode trunk
```

```
user@PE1# set ae1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2
  VLAN3 VLAN4 VLAN5 ]
```

5. Configure IRB interfaces and virtual-gateway addresses.

```
[edit interfaces]
user@PE1# set irb unit 1 family inet address 10.1.1.1/24 virtual-gateway-address
  10.1.1.10
user@PE1# set irb unit 2 family inet address 10.2.1.1/24 virtual-gateway-address
  10.2.1.10
user@PE1# set irb unit 3 family inet address 10.3.1.1/24 virtual-gateway-address
  10.3.1.10
user@PE1# set irb unit 4 family inet address 10.4.1.1/24 virtual-gateway-address
  10.4.1.10
user@PE1# set irb unit 5 family inet address 10.5.1.1/24 virtual-gateway-address
  10.5.1.10
```

6. Configure the autonomous system.

```
[edit routing-options]
user@PE1# set router-id 192.168.1.1
user@PE1# set autonomous-system 65536
```

7. Configure OSPF.

```
[edit protocols ospf]
user@PE1# set area 0.0.0.0 interface xe-0/0/1:0.0
user@PE1# set area 0.0.0.0 interface lo0. passive
```

8. Configure BGP internal peering.

```
[edit protocols bgp]
user@PE1# set group INT type internal
user@PE1# set group INT local-address 192.168.1.1
user@PE1# set group INT family evpn signaling
user@PE1# set group INT local-as 65536
user@PE1# set group INT neighbor 172.16.1.1
```

9. Configure the VLANs.

```
[set vlans]
user@PE1# set VLAN1 vlan-id 1
user@PE1# set VLAN1 l3-interface irb.1
user@PE1# set VLAN1 vxlan vni 1
user@PE1# set VLAN1 vxlan ingress-node-replication
user@PE1# set VLAN2 vlan-id 2
user@PE1# set VLAN2 l3-interface irb.2
user@PE1# set VLAN2 vxlan vni 2
user@PE1# set VLAN2 vxlan ingress-node-replication
```

```
user@PE1# set VLAN3 vlan-id 3
user@PE1# set VLAN3 l3-interface irb.3
user@PE1# set VLAN3 vxlan vni 3
user@PE1# set VLAN4 vxlan ingress-node-replication
user@PE1# set VLAN4 vlan-id 4
user@PE1# set VLAN4 l3-interface irb.4
user@PE1# set VLAN4 vxlan vni 4
user@PE1# set VLAN4 vxlan ingress-node-replication
user@PE1# set VLAN5 vlan-id 5
user@PE1# set VLAN5 l3-interface irb.5
user@PE1# set VLAN2 vxlan vni 5
user@PE1# set VLAN2 vxlan ingress-node-replication
```

10. Enable EVPN.

```
[edit protocols evpn]
user@PE1# set encapsulation vxlan
user@PE1# set multicast-mode ingress-replication
user@PE1# set extended-vni-list 1-5
```

11. Configure an export routing policy to load balance EVPN traffic.

```
[edit policy-options]
user@PE1# set policy-statement evpn-pplb from protocol evpn
user@PE1# set policy-statement evpn-pplb then load-balance per packet

[edit routing-options]
user@PE1# set forwarding-table export evpn-pplb
```

12. Configure the source interface for the VXLAN tunnel.

```
[edit switch-options]
user@PE1# set vtep-source-interface lo0.0
user@PE1# set route-distinguisher 192.168.1.1:1
user@PE1# set vrf-target target:1:1
```

13. Enable IGMP on the IRB interfaces associated with the VLANs..

```
[edit protocols igmp]
user@PE1# set interface irb.1
user@PE1# set interface irb.2
user@PE1# set interface irb.3
user@PE1# set interface irb.4
user@PE1# set interface irb.5
```

14. Enable IGMP snooping on the VLANs.

```
[edit protocols igmp-snooping vlan]
user@PE1# set VLAN1 l2-querier source-address 10.1.1.1
```

```
user@PE1# set VLAN1 proxy
user@PE1# set VLAN2 l2-querier source-address 10.2.1.1
user@PE1# set VLAN2 proxy
user@PE1# set VLAN3 l2-querier source address 10.3.1.1
user@PE1# set VLAN3 proxy
user@PE1# set VLAN4 l2-querier source-address 10.4.1.1
user@PE1# set VLAN4 proxy
user@PE1# set VLAN5 l2-querier source-address 10.5.1.1
user@PE1# set VLAN5 proxy
```

15. Configure PIM by defining a static rendezvous point and enabling on the IRB interfaces associated with the VLANs..

NOTE: This step is required only if you want to configure inter-VLAN forwarding. If your PE device is performing only intra-VLAN forwarding, omit this step.

```
[edit protocols pim]
user@PE1# set rp static address 172.16.1.1
user@PE1# set interface irb.1 distrbuted-dr
user@PE1# set interface irb.2 distributed-dr
user@PE1# set interface irb.3 distributed-dr
user@PE1# set interface irb.4 distributed-dr
user@PE1# set interface irb.5 distributed-dr
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**. **show routing-options**,**show protocols**, **show vlans**, **show policy-options**, and **show switch-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
   ethernet {
      device-count 2;
   }
}
```

```
user@PE1# show
interfaces {
   xe-0/0/0:1 {
      description "Connected to CE";
      ether-options {
         802.3ad ae0;
      }
   }
   xe-0/0/0:2 {
      ether-options {
```

```
      802.3ad ae1;
    }
  }
  xe-0/0/1:0 {
    enable;
    unit 0 {
      description "Connected to PE3";
      family inet {
        address 192.0.2.1/24;
      }
    }
  }
  xe-0/0/1:1 {
    enable;
  }
  ae0 {
    enable;
    esi {
      00:11:11:11:11:11:11:11:11:11;
      all-active;
    }
    aggregated-ether-options {
      lacp {
        active;
        periodic fast;
        system-id 00:00:00:00:00:01;
      }
    }
    unit 0 {
      description "Connected to CE";
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
        }
      }
    }
  }
  ae1 {
    enable;
    esi {
      00:22:22:22:22:22:22:22:22:22;
      all-active;
    }
    aggregated-ether-options {
      lacp {
        active;
        periodic fast;
        system-id 00:00:00:00:00:02;
      }
    }
    unit 0 {
      description "Connected to CE";
      family ethernet-switching {
        interface-mode trunk;
```

```
            vlan {
              members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
            }
          }
        }
      }
      irb {
        unit 1 {
          family inet {
            address 10.1.1.1/24 {
              virtual-gateway-address 10.1.1.10;
            }
          }
        }
        unit 2 {
          family inet {
            address 10.2.1.1./24 {
              virtual-gateway-address 10.2.1.10;
            }
          }
        }
        unit 3 {
          family inet {
            address 10.3.1.1./24 {
              virtual-gateway-address 10.3.1.10;
            }
          }
        }
        unit 4 {
          family inet {
            address 10.4.1.1./24 {
              virtual-gateway-address 10.4.1.10;
            }
          }
        }
        unit 5 {
          family inet {
            address 10.5.1.1./24 {
              virtual-gateway-address 10.5.1.10;
            }
          }
        }
      }
      lo0 {
        unit 0 {
          family inet {
            address 192.168.1.1/32;
          }
        }
      }
}
routing-options {
    router-id 192.168.1.1;
    autonomous-system 65536;
    forwarding-table {
```

```
      export evpn-pplb;
    }
}
protocols {
    igmp {
        interface irb.1;
        interface irb.2;
        interface irb.3;
        interface irb.4;
        interface irb.5;
    }
    bgp {
        group INT {
            type internal;
            local-address 192.168.1.1;
            family evpn {
                signaling;
            }
            local-as 65536;
            neighbor 172.16.1.1;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface xe-0/0/1:0.0;
            interface lo0.0 {
                passive;
            }
        }
    }
    pim {
        rp {
            static {
                address 172.16.1.1;
            }
        }
        interface irb.1 {
            distributed-dr;
        }
        interface irb.2 {
            distributed-dr;
        }
        interface irb.3 {
            distributed-dr;
        }
        interface irb.4 {
            distributed-dr;
        }
        interface irb.5 {
            distributed-dr;
        }
    }
    evpn {
        encapsulation vxlan;
        multicast-mode ingress-replication;
```

```
    extended-vni-list 1-5;
  }
  igmp-snooping {
    vlan VLAN1 {
      l2-querier {
        source-address 10.1.1.1;
      }
      proxy;
    }
    vlan VLAN2 {
      l2-querier {
        source-address 10.2.1.1;
      }
      proxy;
    }
    vlan VLAN3 {
      l2-querier {
        source-address 10.3.1.1;
      }
      proxy;
    }
    vlan VLAN4 {
      l2-querier {
        source-address 10.4.1.1;
      }
      proxy;
    }
    vlan VLAN5 {
      l2-querier {
        source-address 10.5.1.1;
      }
      proxy;
    }
  }
}
policy-options {
  policy-statement evpn-pplb {
    from protocol evpn;
    then {
      load-balance per-packet;
    }
  }
}
switch-options {
  vtep-source-interface lo0.0;
  route-distinguisher 192.168.1.1:1;
  vrf-target target:1:1;
}
vlans {
  VLAN1 {
    vlan-id 1;
    l3-interface irb.1;
    vxlan {
      vni 1;
      ingress-node-replication;
```

```
    }
  }
  VLAN2 {
    vlan-id 2;
    l3-interface irb.2;
    vxlan {
      vni 2;
      ingress-node-replication;
    }
  }
  VLAN3 {
    vlan-id 3;
    l3-interface irb.3;
    vxlan {
      vni 3;
      ingress-node-replication;
    }
  }
  VLAN4 {
    vlan-id 4;
    l3-interface irb.4;
    vxlan {
      vni 4;
      ingress-node-replication;
    }
  }
  VLAN5 {
    vlan-id 5;
    l3-interface irb.5;
    vxlan {
      vni 5;
      ingress-node-replication;
    }
  }
}
```

Configuring PE2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device PE2:

1. Specify the number of aggregated Ethernet logical interfaces.

```
[edit chassis]
user@PE2# set aggregated-devices ethernet device-count 2
```

2. Configure the interfaces.

```
[edit interfaces]
```

```
user@PE2# set xe-0/0/0:0 ether-options 802.3ad ae1
user@PE2# set xe-0/0/0:1 description “Connected to CE”
user@PE2# set xe-0/0/1:0 enable
user@PE2# set xe-0/0/1:0 unit 0 description “Connected to PE3”
user@PE2# set xe-0/0/1:0 unit 0 family inet address 198.51.100.1/24
user@PE2# set ae0 enable
user@PE2# set ae1 enable
user@PE2# set lo0 unit 0 family inet address 192.168.2.1/32
```

3. Configure active-active multihoming and enable the Link Aggregation Control Protocol (LACP) on each aggregated Ethernet interface.

```
[edit interfaces]
user@PE2# set ae0 esi 00:11:11:11:11:11:11:11:11:11
user@PE2# set esi all-active
user@PE2# set ae0 aggregated-ether-options lacp active
user@PE2# set ae0 aggregated-ether-options lacp periodic-fast
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
user@PE2# set ae1 esi 00:22:22:22:22:22:22:22:22:22
user@PE2# set ae1 esi all-active
user@PE2# set ae1 aggregated-ether-options lacp active
user@PE2# set ae1 aggregated-ether-options lacp periodic-fast
user@PE2# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:02
```

4. Configure each aggregated Ethernet interface as a trunk port.

```
[edit interfaces]
user@PE2# set ae0 unit 0 description “Connected to CE”
user@PE2# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@PE2# set ae0 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2
  VLAN3 VLAN4 VLAN5 ]
user@PE2# set ae1 unit 0 description “Connected to CE”
user@PE2# set ae1 unit 0 family ethernet-switching interface-mode trunk
user@PE2# set ae1 unit 0 family ethernet-switching vlan members [ VLAN1 VLAN2
  VLAN3 VLAN4 VLAN5 ]
```

5. Configure IRB interfaces and virtual-gateway addresses.

```
[edit interfaces]
user@PE2# set irb unit 1 family inet address 10.1.1.2/24 virtual-gateway-address
  10.1.1.10
user@PE2# set irb unit 2 family inet address 10.2.1.2/24 virtual-gateway-address
  10.2.1.10
user@PE2# set irb unit 3 family inet address 10.3.1.2/24 virtual-gateway-address
  10.3.1.10
user@PE2# set irb unit 4 family inet address 10.4.1.2/24 virtual-gateway-address
  10.4.1.10
user@PE2# set irb unit 5 family inet address 10.5.1.2/24 virtual-gateway-address
  10.5.1.10
```

6. Configure the autonomous system.

```
[edit routing-options]
user@PE2# set router-id 192.168.2.1
user@PE2# set autonomous-system 65536
```

7. Configure OSPF.

```
[edit protocols ospf]
user@PE2# set area 0.0.0.0 interface xe-0/0/1:0.0
user@PE2# set area 0.0.0.0 interface lo0. passive
```

8. Configure BGP internal peering.

```
[edit protocols bgp]
user@PE2# set group INT type internal
user@PE2# set group INT local-address 192.168.2.1
user@PE2# set group INT family evpn signaling
user@PE2# set group INT local-as 65536
user@PE2# set group INT neighbor 172.16.1.1
```

9. Configure the VLANs.

```
[edit vlans]
user@PE2# set VLAN1 vlan-id 1
user@PE2# set VLAN1 l3-interface irb.1
user@PE2# set VLAN1 vxlan vni 1
user@PE2# set VLAN1 vxlan ingress-node-replication
user@PE2# set VLAN2 vlan-id 2
user@PE2# set VLAN2 l3-interface irb.2
user@PE2# set VLAN2 vxlan vni 2
user@PE2# set VLAN2 vxlan ingress-node-replication
user@PE2# set VLAN3 vlan-id 3
user@PE2# set VLAN3 l3-interface irb.3
user@PE2# set VLAN3 vxlan vni 3
user@PE2# set VLAN4 vxlan ingress-node-replication
user@PE2# set VLAN4 vlan-id 4
user@PE2# set VLAN4 l3-interface irb.4
user@PE2# set VLAN4 vxlan vni 4
user@PE2# set VLAN4 vxlan ingress-node-replication
user@PE2# set VLAN5 vlan-id 5
user@PE2# set VLAN5 l3-interface irb.5
user@PE2# set VLAN2 vxlan vni 5
user@PE2# set VLAN2 vxlan ingress-node-replication
```

10. Enable EVPN.

```
[edit protocols evpn]
user@PE2# set encapsulation vxlan
user@PE2# set multicast-mode ingress-replication
```

```
user@PE2# set extended-vni-list 1-5
```

11. Configure an export routing policy to load balance EVPN traffic and apply it to the forwarding-table.

```
[edit policy-options]
user@PE2# set policy-statement evpn-pplb from protocol evpn
user@PE2# set policy-statement evpn-pplb then load-balance per packet

[edit routing-options]
user@PE2# set forwarding-table export evpn-pplb
```

12. Configure the source interface for the VXLAN tunnel.

```
[edit switch-options]
user@PE2# set vtep-source-interface lo0.0
user@PE2# set route-distinguisher 192.168.2.1:1
user@PE2# set vrf-target target:1:1
```

13. Enable IGMP on the IRB interfaces.

```
[edit protocols igmp]
user@PE2# set interface irb.1
user@PE2# set interface irb.2
user@PE2# set interface irb.3
user@PE2# set interface irb.4
user@PE2# set interface irb.5
```

14. Enable IGMP snooping on the IRB interfaces.

```
[edit protocols igmp-snooping vlan]
user@PE2# set VLAN1 l2-querier source-address 10.1.1.2
user@PE2# set VLAN1 proxy
user@PE2# set VLAN2 l2-querier source-address 10.2.1.2
user@PE2# set VLAN2 proxy
user@PE2# set VLAN3 l2-querier source address 10.3.1.2
user@PE2# set VLAN3 proxy
user@PE2# set VLAN4 l2-querier source-address 10.4.1.2
user@PE2# set VLAN4 proxy
user@PE2# set VLAN5 l2-querier source-address 10.5.1.2
user@PE2# set VLAN5 proxy
```

15. Configure PIM by defining a static rendezvous point and enabling on the IRB interfaces.

NOTE: This step is required only if you want to configure inter-VLAN forwarding. If your PE device is performing only intra-VLAN forwarding, omit this step.

```
[edit protocols pim]
user@PE2# set rp static address 172.16.1.1
user@PE2# set interface irb.1 distrbuted-dr
user@PE2# set interface irb.2 distributed-dr
user@PE2# set interface irb.3 distributed-dr
user@PE2# set interface irb.4 distributed-dr
user@PE2# set interface irb.5 distributed-dr
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**. **show routing-options**,**show protocols**, **show vlans**, **show policy-options**, and **show switch-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show chassis
aggregated-devices {
  ethernet {
    device-count 2;
  }
}
```

```
user@PE2# show
interfaces {
  xe-0/0/0:0 {
    ether-options {
      802.3ad ae1;
    }
  }
  xe-0/0/0:1 {
    description "Connected to CE";
    ether-options {
      802.3ad ae0;
    }
  }
  xe-0/0/1:0 {
    enable;
    unit 0 {
      description "Connected to PE3";
      family inet {
        address 198.51.100.1/24;
      }
    }
  }
  ae0 {
    esi {
```

```
      00:11:11:11:11:11:11:11:11:11;
      all-active;
    }
    aggregated-ether-options {
      lacp {
        active;
        periodic fast;
        system-id 00:00:00:00:00:01;
      }
    }
    unit 0 {
      description "Connected to CE";
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
        }
      }
    }
  }
  ae1 {
    enable;
    esi {
      00:22:22:22:22:22:22:22:22:22;
      all-active;
    }
    aggregated-ether-options {
      lacp {
        active;
        periodic fast;
        system-id 00:00:00:00:00:02;
      }
    }
    unit 0 {
      description "CONNECTED TO CE";
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
        }
      }
    }
  }
  irb {
    unit 1 {
      family inet {
        address 10.1.1.2/24 {
          virtual-gateway-address 10.1.1.10;
        }
      }
    }
    unit 2 {
      family inet {
        address 10.2.1.2/24 {
          virtual-gateway-address 10.2.1.10;
```

```
            }
          }
        }
        unit 3 {
          family inet {
            address 10.3.1.2/24 {
              virtual-gateway-address 10.3.1.10;
            }
          }
        }
        unit 4 {
          family inet {
            address 10.4.1.2/24 {
              virtual-gateway-address 10.4.1.10;
            }
          }
        }
        unit 5 {
          family inet {
            address 10.5.1.2/24 {
              virtual-gateway-address 10.5.1.10;
            }
          }
        }
      }
      lo0 {
        unit 0 {
          family inet {
            address 192.168.2.1/32;
          }
        }
      }
    }
    routing-options {
      router-id 192.168.2.1;
      autonomous-system 65536;
      forwarding-table {
        export evpn-pplb;
      }
    }
    protocols {
      igmp {
        interface irb.1;
        interface irb.2;
        interface irb.3;
        interface irb.4;
        interface irb.5;
      }
      bgp {
        group INT {
          type internal;
          local-address 192.168.2.1;
          family evpn {
            signaling;
          }
```

```
      local-as 65536;
      neighbor 172.16.1.1;
    }
  }
  ospf {
    area 0.0.0.0 {
      interface lo0.0 {
        passive;
      }
      interface xe-0/0/1:0.0;
    }
  }
  pim {
    rp {
      static {
        address 172.16.1.1;
      }
    }
    interface irb.1 {
      distributed-dr;
    }
    interface irb.2 {
      distributed-dr;
    }
    interface irb.3 {
      distributed-dr;
    }
    interface irb.4 {
      distributed-dr;
    }
    interface irb.5 {
      distributed-dr;
    }
  }
  evpn {
    encapsulation vxlan;
    multicast-mode ingress-replication;
    extended-vni-list 1-5;
  }
  igmp-snooping {
    vlan VLAN1 {
      l2-querier {
        source-address 10.1.1.2;
      }
      proxy;
    }
    vlan VLAN2 {
      l2-querier {
        source-address 10.2.1.2;
      }
      proxy;
    }
    vlan VLAN3 {
      l2-querier {
        source-address 10.3.1.2;
```

```
        }
        proxy;
      }
      vlan VLAN4 {
        l2-querier {
          source-address 10.4.1.2;
        }
        proxy;
      }
      vlan VLAN5 {
        l2-querier {
          source-address 10.5.1.2;
        }
        proxy;
      }
    }
}
policy-options {
    policy-statement evpn-pplb {
      from protocol evpn;
      then {
        load-balance per-packet;
      }
    }
}
switch-options {
    vtep-source-interface lo0.0;
    route-distinguisher 192.168.2.1:1;
    vrf-target target:1:1;
}
vlans {
    VLAN1 {
      vlan-id 2;
      l3-interface irb.2;
      vxlan {
        vni 2;
        ingress-node-replication;
      }
    }
    VLAN2 {
      vlan-id 1;
      l3-interface irb.1;
      vxlan {
        vni 1;
        ingress-node-replication;
      }
    }
    VLAN3 {
      vlan-id 3;
      l3-interface irb.3;
      vxlan {
        vni 3;
        ingress-node-replication;
      }
    }
```

```
    VLAN4 {
      vlan-id 4;
      l3-interface irb.4;
      vxlan {
        vni 4;
        ingress-node-replication;
      }
    }
    VLAN5 {
      vlan-id 5;
      l3-interface irb.5;
      vxlan {
        vni 5;
        ingress-node-replication;
      }
    }
  }
```

Configuring CE Device

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device CE:

1. Specify the number of aggregated Ethernet logical interfaces.

```
[edit chassis]
user@CE# set aggregated-devices ethernet device-count 2
```

2. Configure the interfaces and enable LACP on the aggregated Ethernet interfaces.

```
[edit interfaces]
user@CE# set xe-0/2/0 ether-options 802.3ad ae0
user@CE# set xe-0/2/1 ether-options 802.3ad ae0
user@CE# set ae0 aggregated ether-options lacp active
user@CE# set ae0 aggregated ether-options lacp periodic fast
user@CE# set ae1 aggregated ether-options lacp active
user@CE# set ae1 aggregated ether-options lacp periodic fast
```

3. Create the Layer 2 customer bridge domains and the VLANs associated with the domains.

```
[edit]
user@CE# set vlans BD-1 domain-type bridge
user@CE# set vlans BD-1 vlan-id 1
user@CE# set vlans BD-2 domain-type bridge
user@CE# set vlans BD-2 vlan-id 2
user@CE# set vlans BD-3 domain-type bridge
user@CE# set vlans BD-3 vlan-id 3
```

```
user@CE# set vlans BD-4 domain-type bridge
user@CE# set vlans BD-4 vlan-id 4
user@CE# set vlans BD-5 domain-type bridge
user@CE# set vlans BD-5 vlan-id 5
```

4. Configure each interface to include in CE domain as a trunk port for accepting packets tagged with the specified VLAN identifiers.

```
[edit interfaces]
user@CE# set xe-0/2/0 unit 0 family ethernet-switching interface-mode trunk
user@CE# set xe-0/2/0 unit 0 family ethernet-switching vlan members 1-5
user@CE# set xe-0/2/1 unit 0 family ethernet-switching interface-mode trunk
user@CE# set xe-0/2/1 unit 0 family ethernet-switching vlan members 1-5
user@CE# set ae0 unit 0 family ethernet-switching interface-mode trunk
user@CE# set ae0 unit 0 family ethernet-switching vlan members 1-5
user@CE# set ae1 unit 0 family ethernet-switching interface-mode trunk
user@CE# set ae1 unit 0 family ethernet-switching vlan members 1-5
```

Results From configuration mode, confirm your configuration by entering the **show chassis** and **show interfaces** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE# show chassis
aggregated-devices {
  ethernet {
    device-count 2;
  }
}
```

```
user@CE# show interfaces
xe-0/2/0 {
  ether-options {
    802.3ad ae0;
  }
  unit 0 {
    family ethernet-switching {
      interface-mode trunk;
      vlan {
        members 1-5;
      }
    }
  }
}
xe-0/2/1 {
  ether-options {
    802.3ad ae0;
  }
  unit 0 {
    family ethernet-switching {
      interface-mode trunk;
```



```
        vlan {
          members 1-5;
        }
      }
    }
}
ae0 {
    aggregated-ether-options {
      lacp {
        active;
        periodic fast;
      }
    }
    unit 0 {
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members 1-5;
        }
      }
    }
}
ae1 {
    aggregated-ether-options {
      lacp {
        active;
        periodic fast;
      }
    }
    unit 0 {
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members 1-5;
        }
      }
    }
}
```

```
user@CE# show vlans
BD-1 {
    vlan-id 1;
    domain-type bridge;
}
BD-2 {
    vlan-id 2;
    domain-type bridge;
}
BD-3 {
    vlan-id 3;
    domain-type bridge;
}
BD-3 {
    vlan-id 3;
```

```
    domain-type bridge;
}
BD-4 {
    vlan-id 4;
    domain-type bridge;
}
BD-5 {
    vlan-id 5;
    domain-type bridge;
}
```

Configuring PE3

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure device PE3:

1. Configure the interfaces.

```
[edit interfaces]
user@PE3# set xe-0/0/0 enable
user@PE3# set xe-0/0/0 unit 0 description “Connected to PE1”
user@PE3# set xe-0/0/0 unit 0 family inet 192.0.2.2/24
user@PE3# set xe-0/0/1 unit 0 description “Connected to PE2”
user@PE3# set xe-0/0/1 198.51.100.2/24
user@PE3# set lo0 unit 0 family inet address 172.16.1.1/32
```

2. Configure each logical Ethernet interface as a trunk port for accepting packets tagged with the specified VLAN identifiers.

```
[edit interfaces]
user@PE3# set xe-0/0/0:1 enable
user@PE3# set xe-0/0/0:1 unit 0 family ethernet-switching interface-mode trunk
user@PE3# set xe-0/0/0:1 unit 0 family ethernet-switching vlan members [ VLAN1
    VLAN2 VLAN3 VLAN4 VLAN5 ]
user@PE3# set xe-0/0/1:1 enable
user@PE3# set xe-0/0/1:1 unit 0 family ethernet-switching interface-mode trunk
user@PE3# set xe-0/0/1:1 unit 0 family ethernet-switching vlan members [ VLAN1
    VLAN2 VLAN3 VLAN4 VLAN5 ]
```

3. Configure IRB interfaces and virtual-gateway addresses.

```
[edit interfaces]
user@PE3# set irb unit 1 family inet address 10.1.1.5/24 virtual-gateway-address
    10.1.1.10
user@PE3# set irb unit 2 family inet address 10.2.1.5/24 virtual-gateway-address
    10.2.1.10
user@PE3# set irb unit 3 family inet address 10.3.1.5/24 virtual-gateway-address
    10.3.1.10
```

```
user@PE3# set irb unit 4 family inet address 10.4.1.5/24 virtual-gateway-address
  10.4.1.10
user@PE3# set irb unit 5 family inet address 10.5.1.5/24 virtual-gateway-address
  10.5.1.10
```

4. Configure the autonomous system.

```
[edit routing-options]
user@PE3# set router-id 172.16.1.1
user@PE3# set autonomous-system 65536
```

5. Configure OSPF

```
[edit protocols ospf]
user@PE3# set area 0.0.0.0 interface all
user@PE3# set ospf area 0.0.0.0 fxp0.0 disable
```

6. Configure BGP internal peering with PE1 and PE2.

```
[edit protocols bgp]
user@PE3# set group INT type internal
user@PE3# set group INT local-address 172.16.1.1
user@PE3# set group INT family evpn signaling
user@PE3# set group INT local-as 65536
user@PE3# set group INT neighbor 192.168.1.1
user@PE3# set group INT neighbor 192.168.2.1
```

7. Configure the VLANs.

```
[edit vlans]
user@PE3# set VLAN1 vlan-id 1
user@PE3# set VLAN1 l3-interface irb.1
user@PE3# set VLAN1 vxlan vni 1
user@PE13# set VLAN1 vxlan ingress-node-replication
user@PE3# set VLAN2 vlan-id 2
user@PE3# set VLAN2 l3-interface irb.2
user@PE3# set VLAN2 vxlan vni 2
user@PE3# set VLAN2 vxlan ingress-node-replication
user@PE3# set VLAN3 vlan-id 3
user@PE3# set VLAN3 l3-interface irb.3
user@PE3# set VLAN3 vxlan vni 3
user@PE3# set VLAN4 vxlan ingress-node-replication
user@PE3# set VLAN4 vlan-id 4
user@PE3# set VLAN4 l3-interface irb.4
user@PE3# set VLAN4 vxlan vni 4
user@PE3# set VLAN4 vxlan ingress-node-replication
user@PE3# set VLAN5 vlan-id 5
user@PE3# set VLAN5 l3-interface irb.5
user@PE3# set VLAN2 vxlan vni 5
user@PE3# set VLAN2 vxlan ingress-node-replication
```

8. Enable EVPN.

```
[edit protocols evpn]
user@PE3# set encapsulation vxlan
user@PE3# set multicast-mode ingress-replication
user@PE3# set extended-vni-list 1-5
```

9. Configure an export routing policy to load balance EVPN traffic.

```
[edit policy-options]
user@PE3# set policy-statement evpn-pplb from protocol evpn
user@PE3# set policy-statement evpn-pplb then load-balance per packet

[edit routing-options]
user@PE3# set forwarding-table export evpn-pplb
```

10. Configure the source interface for the VXLAN tunnel.

```
[edit switch-options]
user@PE3# set vtep-source-interface lo0.0
user@PE3# set route-distinguisher 172.16.1.1:1
user@PE3# set vrf-target target:1:1
```

11. Enable IGMP on the IRB interfaces.

```
[edit protocols igmp]
user@PE1# set interface irb.1
user@PE1# set interface irb.2
user@PE1# set interface irb.3
user@PE1# set interface irb.4
user@PE1# set interface irb.5
```

12. Enable IGMP snooping on the IRB interfaces.

```
[edit protocols igmp-snooping vlan]
user@PE1# set VLAN1 l2-querier source-address 10.1.1.5
user@PE1# set VLAN1 proxy
user@PE1# set VLAN2 l2-querier source-address 10.2.1.5
user@PE1# set VLAN2 proxy
user@PE1# set VLAN3 l2-querier source address 10.3.1.5
user@PE1# set VLAN3 proxy
user@PE1# set VLAN4 l2-querier source-address 10.4.1.5
user@PE1# set VLAN4 proxy
user@PE1# set VLAN5 l2-querier source-address 10.5.1.5
user@PE1# set VLAN5 proxy
```

13. Configure PIM by defining the local rendezvous point and enabling on the IRB interfaces.

NOTE: This step is required only if you want to configure inter-VLAN forwarding. If your PE device is performing only intra-VLAN forwarding, omit this step.

```
[edit protocols pim]
user@PE1# set rp local address 172.16.1:1
user@PE1# set interface irb.1 distrbuted-dr
user@PE1# set interface irb.2 distributed-dr
user@PE1# set interface irb.3 distributed-dr
user@PE1# set interface irb.4 distributed-dr
user@PE1# set interface irb.5 distributed-dr
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**. **show routing-options**,**show protocols**, **show vlans**, **show policy-options**, and **show switch-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show
interfaces {
   xe-0/0/0 {
      enable;
      unit 0 {
         description "Connected to PE1";
         family inet {
            address 192.0.2.2/24;
         }
      }
   }
   xe-0/0/1 {
      enable;
      unit 0 {
         description “Connected to PE2”;
         family inet {
            address 198.51.100.2/24
         }
      }
   }
   xe-0/0/0:1 {
      enable;
      unit 0 {
         family ethernet-switching {
            interface-mode trunk;
            vlan {
               members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
            }
         }
      }
   }
   xe-0/0/1:1 enable;
```

```
    unit 0 {
      family ethernet-switching {
        interface-mode trunk;
        vlan {
          members [ VLAN1 VLAN2 VLAN3 VLAN4 VLAN5 ];
        }
      }
    }
  }
  irb {
    unit 1 {
      family inet {
        address 10.1.1.5/24 {
          virtual-gateway-address 10.1.1.10;
        }
      }
    }
    unit 2 {
      family inet {
        address 10.2.1.5/24 {
          virtual-gateway-address 10.2.1.10;
        }
      }
    }
    unit 3 {
      family inet {
        address 10.3.1.5/24 {
          virtual-gateway-address 10.3.1.10;
        }
      }
    }
    unit 4 {
      family inet {
        address 10.4.1.5/24 {
          virtual-gateway-address 10.4.1.10;
        }
      }
    }
    unit 5 {
      family inet {
        address 10.5.1.5/24 {
          virtual-gateway-address 10.5.1.10;
        }
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 172.16.1.1/32;
      }
    }
  }
}
routing-options {
```

```
    router-id 172.16.1.1;
    autonomous-system 65536
    forwarding-table {
      export evpn-pplb;
    }
}
protocols {
    igmp {
      interface irb.1;
      interface irb.2;
      interface irb.3;
      interface irb.4;
      interface irb.5;
    }
    bgp {
      group INT {
        type internal;
        local-address 172.16.1.1;
        family evpn {
          signaling;
        }
        local-as 65546;
        neighbor 192.168.1.1;
        neighbor 192.168.2.1;
      }
    }
    ospf {
      area 0.0.0.0 {
        interface all ;
        interface fxp0.0;
          disable;
        }
      }
    }
    pim {
      rp {
        local {
          address 172.16.1.1;
        }
      }
      interface irb.1 {
        distributed-dr;
      }
      interface irb.2 {
        distributed-dr;
      }
      interface irb.3 {
        distributed-dr;
      }
      interface irb.4 {
        distributed-dr;
      }
      interface irb.5 {
        distributed-dr;
      }
```

```
    }
    evpn {
      encapsulation vxlan;
      multicast-mode ingress-replication;
      extended-vni-list 1-5;
    }
    igmp-snooping {
      vlan VLAN1 {
        l2-querier {
          source-address 10.1.1.5;
        }
        proxy;
      }
      vlan VLAN2 {
        l2-querier {
          source-address 10.2.1.5;
        }
        proxy;
      }
      vlan VLAN3 {
        l2-querier {
          source-address 10.3.1.5;
        }
        proxy;
      }
      vlan VLAN4 {
        l2-querier {
          source-address 10.4.1.5;
        }
        proxy;
      }
      vlan VLAN5 {
        l2-querier {
          source-address 10.5.1.5;
        }
        proxy;
      }
    }
}
policy-options {
    policy-statement evpn-pplb {
      from protocol evpn;
      then {
        load-balance per-packet;
      }
    }
}
switch-options {
    vtep-source-interface lo0.0;
    route-distinguisher 172.16.1.1:1;
    vrf-target target:1:1;
}
vlans {
    VLAN1 {
      vlan-id 1;
```

```
        l3-interface irb.1;
        vxlan {
          vni 1;
          ingress-node-replication;
        }
      }
      VLAN2 {
        vlan-id 2;
        l3-interface irb.2;
        vxlan {
          vni 2;
          ingress-node-replication;
        }
      }
      VLAN3 {
        vlan-id 3;
        l3-interface irb.3;
        vxlan {
          vni 3;
          ingress-node-replication;
        }
      }
      VLAN4 {
        vlan-id 4;
        l3-interface irb.4;
        vxlan {
          vni 4;
          ingress-node-replication;
        }
      }
      VLAN5 {
        vlan-id 5;
        l3-interface irb.5;
        vxlan {
          vni 5;
          ingress-node-replication;
        }
      }
    }
```

Verification

Confirm that the configuration is working properly.

- Verifying IGMP Messages are Synced on page 521
- Verifying Source Addresses Are Learned and Multicast Traffic Is Being Forwarded on page 523

Verifying IGMP Messages are Synced

Purpose Verify on each PE that IGMP join and leave messages are synced.

Action From operational mode, run the **show evpn instance extensive** command.

```
user@PE1> show evpn instance extensive

Instance: default-switch
  Route Distinguisher: 192.168.1.1:1
  Encapsulation type: VXLAN
  MAC database status                       Local  Remote
    MAC advertisements:                       25      40
    MAC+IP advertisements:                    10      15
    Default gateway MAC advertisements:       10      10
  Number of local interfaces: 3 (3 up)
    Interface name  ESI                               Mode             Status
AC-Role
    ae0.0           00:11:11:11:11:11:11:11:11:11  all-active       Up
Root
    ae1.0           00:22:22:22:22:22:22:22:22:22  all-active       Up
Root
    xe-0/0/1:1.0    00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
  Number of IRB interfaces: 5 (5 up)
    Interface name  VLAN   VNI    Status  L3 context
    irb.1                  1       Up     master
    irb.2                  2       Up     master
    irb.3                  3       Up     master
    irb.4                  4       Up     master
    irb.5                  5       Up     master
  Number of bridge domains: 5
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label  SG sync  IM core nexthop
    1     1              3    3     irb.1      Extended         Enabled   1
          Enabled   2097159
    2     2              3    3     irb.2      Extended         Enabled   2
          Enabled   2097161
    3     3              3    3     irb.3      Extended         Enabled   3
          Enabled   2097162
    4     4              3    3     irb.4      Extended         Enabled   4
          Enabled   2097163
    5     5              3    3     irb.5      Extended         Enabled   5
          Enabled   2097164
  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    192.168.2.1            25        10         9         5         0
    172.16.1.1             15        10         1         5         0
  Number of ethernet segments: 7
    ESI: 00:11:11:11:11:11:11:11:11:11
      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
        192.168.2.1      5          0               all-active
      DF Election Algorithm: MOD based
      Designated forwarder: 192.168.2.1
      Backup forwarder: 192.168.1.1
      Last designated forwarder update: Jul 13 01:22:45
    ESI: 00:22:22:22:22:22:22:22:22:22
      Status: Resolved by IFL ae1.0
      Local interface: ae1.0, Status: Up/Forwarding
      Number of remote PEs connected: 1
```

```
      Remote PE        MAC label  Aliasing label  Mode
      192.168.2.1       5          0               all-active
    DF Election Algorithm: MOD based
    Designated forwarder: 192.168.2.1
    Backup forwarder: 192.168.1.1
    Last designated forwarder update: Jul 13 21:02:47
  ESI: 05:00:00:00:64:00:00:00:01:00
    Local interface: irb.1, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE        MAC label  Aliasing label  Mode
      192.168.2.1      1          0               all-active
      172.16.1.1       1          0               single-homed
  ESI: 05:00:00:00:64:00:00:00:02:00
    Local interface: irb.2, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE        MAC label  Aliasing label  Mode
      192.168.2.1      2          0               all-active
      172.16.1.1       2          0               single-homed
  ESI: 05:00:00:00:64:00:00:00:03:00
    Local interface: irb.3, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE        MAC label  Aliasing label  Mode
      192.168.2.1      3          0               all-active
      172.16.1.1       3          0               single-homed
  ESI: 05:00:00:00:64:00:00:00:04:00
    Local interface: irb.4, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE        MAC label  Aliasing label  Mode
      192.168.2.1      4          0               all-active
      172.16.1.1      4          0               single-homed
  ESI: 05:00:00:00:64:00:00:00:05:00
    Local interface: irb.5, Status: Up/Forwarding
    Number of remote PEs connected: 2
      Remote PE        MAC label  Aliasing label  Mode
      172,168.1.1      5          0               all-active
      192.168.2.1      5          0               all-active
Router-ID: 192.168.1.1
```

Meaning The **SG Sync** is **Enabled** and the **IM Core next-hop** field displays a valid route.

Verifying Source Addresses Are Learned and Multicast Traffic Is Being Forwarded

Purpose Verify on each PE that multicast receivers have learned the source interface for the VXLAN tunnel.

Action From operational mode, enter the **show evpn igmp-snooping database extensive l2-domain-id 1** command and the **show igmp snooping evpn database vlan VLAN1** commands.

These commands displays output for VLAN1. You can use them to display output for each configured VLAN

From operational mode, enter the **show evpn multicast-snooping next-hops** to verify that downstream interface has been learned.

```
user@PE1> show evpn igmp-snooping database extensive l2-domain-id 1

Instance: default-switch
  VN Identifier: 1
        Group IP: 225.1.1.1
            Access OIF Count: 2
                  Interface                 ESI             Local    Remote
                ae1.0         00:22:22:22:22:22:22:22:22:22 1          0
                ae0.0         00:11:11:11:11:11:11:11:11:11 0          1
            Remote OIF Count: 2, Core NH: 2097159
                 Interface           Nbr
                vtep.32770    192.168.2.1
                vtep.32769    172.16.1.1
        Group IP: 225.1.1.2
            Access OIF Count: 2
                  Interface                 ESI             Local    Remote
                ae1.0         00:22:22:22:22:22:22:22:22:22 1          0
                ae0.0         00:11:11:11:11:11:11:11:11:11 0          1
            Remote OIF Count: 2, Core NH: 2097159
                 Interface           Nbr
                vtep.32770    192.168.2.1
                vtep.32769    172.16.1.1
        Group IP: 225.1.1.3
            Access OIF Count: 2
                  Interface                 ESI             Local    Remote
                ae0.0         00:11:11:11:11:11:11:11:11:11 1          0
                ae1.0         00:22:22:22:22:22:22:22:22:22 1          0
            Remote OIF Count: 2, Core NH: 2097159
                 Interface           Nbr
                vtep.32770    192.168.2.1
                vtep.32769    172.16.1.1
        Group IP: 225.1.1.4
            Access OIF Count: 2
                  Interface                 ESI             Local    Remote
                ae0.0         00:11:11:11:11:11:11:11:11:11 1          0
                ae1.0         00:22:22:22:22:22:22:22:22:22 0          1
            Remote OIF Count: 2, Core NH: 2097159
                 Interface           Nbr
                vtep.32770    192.168.2.1
                vtep.32769    172.16.1.1
        Group IP: 225.1.1.5
            Access OIF Count: 2
                  Interface                 ESI             Local    Remote
                ae0.0         00:11:11:11:11:11:11:11:11:11 1          0
                ae1.0         00:22:22:22:22:22:22:22:22:22 1          0
            Remote OIF Count: 2, Core NH: 2097159
                 Interface           Nbr
                vtep.32770    192.168.2.1
                vtep.32769    172.16.1.1
        Group IP: 225.1.1.6
            Access OIF Count: 2
                  Interface                 ESI             Local    Remote
                ae0.0         00:11:11:11:11:11:11:11:11:11 0          1
                ae1.0         00:22:22:22:22:22:22:22:22:22 1          0
            Remote OIF Count: 2, Core NH: 2097159
                 Interface           Nbr
                vtep.32770    192.168.2.1
```

```
            vtep.32769    172.16.1.1
      Group IP: 225.1.1.7
          Access OIF Count: 2
                Interface                 ESI             Local    Remote
            ae0.0         00:11:11:11:11:11:11:11:11:11 0          1
            ae1.0         00:22:22:22:22:22:22:22:22:22 1          0
          Remote OIF Count: 2, Core NH: 2097159
               Interface          Nbr
            vtep.32770    192.168.2.1
            vtep.32769    172.16.1.1
      Group IP: 225.1.1.8
          Access OIF Count: 2
                Interface                 ESI             Local    Remote
            ae0.0         00:11:11:11:11:11:11:11:11:11 1          0
            ae1.0         00:22:22:22:22:22:22:22:22:22 0          1
          Remote OIF Count: 2, Core NH: 2097159
               Interface          Nbr
            vtep.32770    192.168.2.1
            vtep.32769    172.16.1.1
      Group IP: 225.1.1.9
          Access OIF Count: 2
                Interface                 ESI             Local    Remote
            ae0.0         00:11:11:11:11:11:11:11:11:11 1          0
            ae1.0         00:22:22:22:22:22:22:22:22:22 0          1
          Remote OIF Count: 2, Core NH: 2097159
               Interface          Nbr
            vtep.32770    192.168.2.1
            vtep.32769    172.16.1.1
      Group IP: 225.1.1.10
          Access OIF Count: 2
                Interface                 ESI             Local    Remote
            ae0.0         00:11:11:11:11:11:11:11:11:11 0          1
            ae1.0         00:22:22:22:22:22:22:22:22:22 0          1
          Remote OIF Count: 2, Core NH: 2097159
               Interface          Nbr
               vtep.32770    192.168.2.1
               vtep.32769    172.16.1.1
```

user@PE1> **show igmp snooing evpn database vlan VLAN1**

```
Instance: default-switch
  Bridge-Domain: VLAN1, VN Identifier: 1
        Group IP: 225.1.1.1
            Core NH: 2097159
            Access OIF Count: 3
                  Interface    Local    Remote
              ae0.0              0        1
              xe-0/0/1:1.0       1        0
              ae1.0              1        0
        Group IP: 225.1.1.2
            Core NH: 2097159
            Access OIF Count: 3
                  Interface    Local    Remote
              ae0.0              0        1
              xe-0/0/1:1.0       1        0
              ae1.0              1        0
        Group IP: 225.1.1.3
```

```
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        xe-0/0/1:1.0     1        0
        ae1.0            1        0
        ae0.0            1        0
Group IP: 225.1.1.4
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        ae1.0            0        1
        xe-0/0/1:1.0     1        0
        ae0.0            1        0
Group IP: 225.1.1.5
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        ae1.0            1        0
        xe-0/0/1:1.0     1        0
        ae0.0            1        0
Group IP: 225.1.1.6
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        ae0.0            0        1
        ae1.0            1        0
        xe-0/0/1:1.0     1        0
Group IP: 225.1.1.7
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        ae0.0            0        1
        ae1.0            1        0
        xe-0/0/1:1.0     1        0
Group IP: 225.1.1.8
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        ae1.0            0        1
        xe-0/0/1:1.0     1        0
        ae0.0            1        0
Group IP: 225.1.1.9
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        ae1.0            0        1
        xe-0/0/1:1.0     1        0
        ae0.0            1        0
Group IP: 225.1.1.10
    Core NH: 2097159
    Access OIF Count: 3
          Interface    Local    Remote
        ae1.0            0        1
        ae0.0            0        1
        xe-0/0/1:1.0     1        0
```

```
user@PE1> show evpn multicast-snooping next-hops
```

```
Family: INET
ID          Refcount KRefcount Downstream interface Addr
2097159            3         1 vtep.32769
                               vtep.32770
2097161            3         1 vtep.32769
                               vtep.32770
2097162            3         1 vtep.32769
                               vtep.32770
2097163            3         1 vtep.32769
                               vtep.32770
2097164            3         1 vtep.32769
                               vtep.32770
```

Related Documentation

- Overview of IGMP Snooping in an EVPN-VXLAN Environment on page 485

CHAPTER 15

Configuring the Tunneling of Q-in-Q Traffic

Examples: Tunneling Q-in-Q Traffic in an EVPN-VXLAN Overlay Network

The tunneling of Q-in-Q packets in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) overlay network is supported as follows:

- Starting with Junos OS Release 17.2R1, QFX5100 switches that function as Layer 2 VXLAN tunnel endpoints (VTEPs) can tunnel single- and double-tagged Q-in-Q packets in an EVPN-VXLAN overlay network with a two-layer IP fabric, which is also known as a *centrally-routed bridging overlay*.
- Starting with Junos OS Release 18.2R1, QFX5110, QFX5200, and EX4600 switches that function as Layer 2 VTEPs can tunnel single- and double-tagged Q-in-Q packets in an EVPN-VXLAN overlay network with a two-layer IP fabric.
- Starting with Junos OS Release 18.3R1, the QFX10000 line of switches that function as Layer 2 VTEPs only can tunnel single- and double-tagged Q-in-Q packets in an EVPN-VXLAN overlay network with a two-layer IP fabric. Also, the QFX10000 line of switches that function as Layer 2 and 3 VTEPs can tunnel single- and double-tagged Q-in-Q packets in an EVPN-VXLAN overlay network with a collapsed IP fabric, which is also known as an *edge-routed bridging overlay*.

In addition to tunneling Q-in-Q packets, the ingress and egress VTEPs can perform the following Q-in-Q actions:

- Delete, or pop, an outer service VLAN (S-VLAN) tag from an incoming packet.
- Add, or push, an outer S-VLAN tag onto an outgoing packet.
- Map a configured range of customer VLAN (C-VLAN) IDs to an S-VLAN.

NOTE: The QFX Series and EX4600 switches support the pop and push actions only with a specified VLAN. The switches do not support the pop and push actions with a configured range of VLANs.

The ingress and egress VTEPs support the tunneling of Q-in-Q packets and the Q-in-Q actions in the context of the traffic patterns described in this topic.

NOTE: This topic describes and shows how to configure the VXLAN tunneling of Q-in-Q packets for each traffic pattern. One or more of the traffic patterns might apply to your environment. Perform only those configurations that apply to your environment.

The ingress and egress VTEPs can also map a single- or double-tagged packet to a specified VLAN or to any VLAN specified in a configured list, and further map the VLAN to a VXLAN network identifier (VNI).

To enable the tunneling of Q-in-Q packets, you must configure a flexible VLAN tagging interface that can transmit 802.1Q VLAN single- and double-tagged packets on ingress and egress VTEPs. Also, on all supported Juniper Networks switches except for the QFX10000 line of switches, it is also important to configure the interface to retain the inner C-VLAN tag while a packet is tunneled. By default, the QFX10000 line of switches retain the inner C-VLAN tag while a packet is tunneled.

This topic includes the following sections:

Requirements

These examples use the following hardware and software components:

- Two QFX5100 switches. One switch functions as the ingress VTEP; and the other as the egress VTEP.
- Junos OS Release 17.2R1 or later.

Overview and Topology

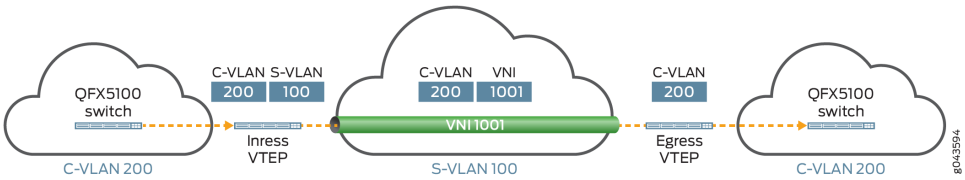
This section describes the traffic patterns in which the VXLAN tunneling of Q-in-Q traffic is supported in a EVPN-VXLAN overlay network.

NOTE: This topic describes and shows how to configure the VXLAN tunneling of Q-in-Q packets for each traffic pattern. One or more of the traffic patterns might apply to your environment. Perform only those configurations that apply to your environment.

Understanding Traffic Pattern 1: Popping an S-VLAN Tag

Figure 47 on page 531 shows Q-in-Q traffic flowing from one dispersed C-VLAN 200 site to another by way of S-VLAN 100.

Figure 47: Popping an S-VLAN Tag



When a packet flows from C-VLAN 200 to S-VLAN 100 to C-VLAN 200, the ingress VTEP:

- Receives a packet with two tags—an inner C-VLAN tag of 200 and an outer S-VLAN tag of 100.
- Takes note of S-VLAN tag 100, which is mapped to VNI 1001, and then pops the tag.
- Encapsulates the packet with a VXLAN header that includes VNI 1001, and sends the packet with inner C-VLAN tag 200 and the VXLAN header.

After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- Removes the VXLAN header from the packet.
- Maps VNI 1001 back to S-VLAN 100.
- Sends the packet with C-VLAN tag 200.

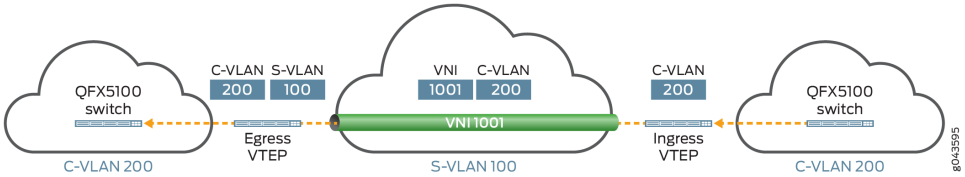
See Also

- Configuring Traffic Pattern 1: Popping an S-VLAN Tag on page 534

Understanding Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN, and Pushing an S-VLAN Tag

Figure 48 on page 532 shows Q-in-Q traffic flowing from one dispersed C-VLAN 200 site to another by way of S-VLAN 100.

Figure 48: Mapping a Range of C-VLANs to an S-VLAN



When a single-tagged packet flows from C-VLAN 200 to S-VLAN 100 to C-VLAN 200, the ingress VTEP:

- Receives a packet with a C-VLAN tag of 200.
- Takes note of C-VLAN tag 200, which is in a configured VLAN ID range 100 through 200 that is mapped to S-VLAN 100 and VNI 1001.
- Encapsulates the packet with a VXLAN header that includes VNI 1001 and sends the packet with C-VLAN tag 200 and VNI 1001.

After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- De-encapsulates the packet.
- Maps the packet to S-VLAN 100 through its association with VNI 1001.
- Pushes S-VLAN tag 100 on the packet, and sends the packet with inner C-VLAN tag 200 and outer S-VLAN tag 100.

See Also

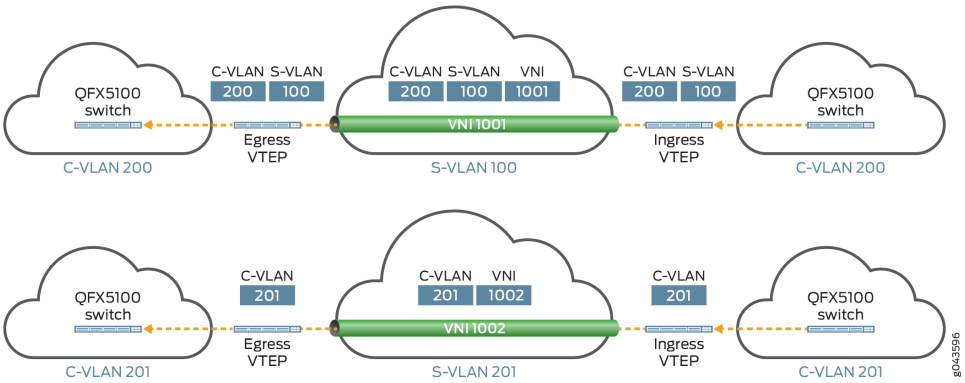
- Configuring Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN on page 537

Understanding Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags

Figure 49 on page 533 shows the following Q-in-Q traffic flows:

- Double-tagged packets from C-VLAN 200 to S-VLAN 100 to C-VLAN 200.
- Single-tagged packets from C-VLAN 201 to S-VLAN 201 to C-VLAN 201.

Figure 49: Retaining S-VLAN and C-VLAN Tags



When a packet flows from either C-VLAN 200 or C-VLAN 201, the ingress VTEP:

- Receives a packet—either a double-tagged packet with an inner C-VLAN tag of 200 and an outer S-VLAN tag of 100 or a single-tagged packet with a C-VLAN tag of 201.
- Takes note of outer S-VLAN tag 100, which is mapped to VNI 1001, for the double-tagged packet. For the single-tagged packet, the ingress VTEP takes note of C-VLAN tag 201, which is mapped to VNI 1002.
- Encapsulates the packet with a VXLAN header that includes VNI 1001 for the double-tagged packet and VNI 1002 for the single-tagged packet. In addition to the VXLAN header, the ingress VTEP sends the double-tagged packet with inner C-VLAN tag 200 and outer S-VLAN tag 100, and the single-tagged packet with C-VLAN tag 201.

After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- Removes the VXLAN header from the packet.
- For the double-tagged packet, maps VNI 1001 back to S-VLAN 100, and for the single-tagged packet, maps VNI 1002 back to C-VLAN 201.
- Sends the double-tagged packet with inner C-VLAN tag 200 and outer S-VLAN tag 100 and the single-tagged packet with C-VLAN tag 201.

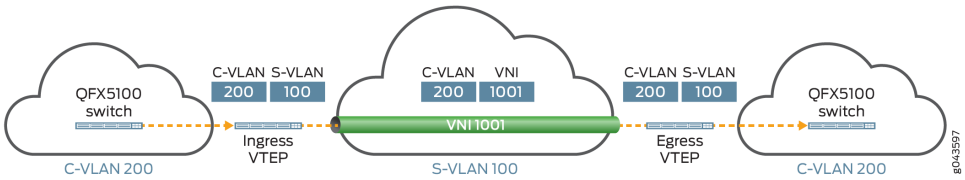
See Also

- Configuring Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags on page 541

Understanding Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag

Figure 50 on page 534 shows Q-in-Q traffic flowing from one dispersed C-VLAN 200 site to another by way of S-VLAN 100.

Figure 50: Popping and Later Pushing an S-VLAN Tag



When a packet flows from C-VLAN 200 to S-VLAN 100 to C-VLAN 200, the ingress VTEP:

- Receives a packet with two tags—an inner C-VLAN tag of 200 and an outer S-VLAN tag of 100.
- Takes note of S-VLAN tag 100, which is mapped to VNI 1001, then pops the tag.
- Encapsulates the packet with a VXLAN header that includes VNI 1001 and sends the packet with inner C-VLAN tag 200 and the VXLAN header.

After the packet is tunneled over the Layer 3 underlay network, the egress VTEP:

- De-encapsulates the packet.
- Maps the packet back to S-VLAN 100 through its association with VNI 1001.
- Pushes S-VLAN tag 100 on the packet, and sends the packet with inner C-VLAN tag 200 and outer S-VLAN tag 100.

See Also

- Configuring Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag on page 543

Configuring Traffic Pattern 1: Popping an S-VLAN Tag

Introduction

For this traffic pattern, the ingress and egress VTEPs in an EVPN-VXLAN overlay network must handle double-tagged Q-in-Q traffic. The ingress VTEP retains the inner C-VLAN tag and removes, or pops, the outer S-VLAN tag. The egress VTEP also retains the inner C-VLAN tag but does not reinstate the outer S-VLAN tag.

NOTE: QFX Series and EX4600 switches support this traffic pattern on both aggregated Ethernet and non-aggregated Ethernet interfaces.

NOTE: This configuration focuses on traffic pattern 1 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a two-layer IP fabric, see “Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center” on page 302.

Ingress VTEP Configuration for Traffic Pattern 1

CLI Quick Configuration

To quickly configure the ingress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

NOTE: You do not need to issue the **set protocols l2-learning decapsulate-accept-inner-vlan** command on the QFX10000 line of switches because, by default, these switches retain the inner C-VLAN tag while a packet is tunneled.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set interfaces xe-0/0/0 unit 100 input-vlan-map pop
set interfaces xe-0/0/0 unit 100 output-vlan-map push
set vlans vlan_1 interface xe-0/0/0.100
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
set vlans vlan_1 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure the ingress VTEP for traffic pattern 1:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

   ```
   [edit protocols l2-learning]
   user@switch# set decapsulate-accept-inner-vlan
   ```

   NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying Tag Protocol Identifier (TPID) 0x8100.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/0 flexible-vlan-tagging
   user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
   ```

3. On physical interface xe-0/0/0, create logical interface 100, and associate it with S-VLAN 100. Also, assuming that the ingress VTEP receives a double-tagged packet as described in this traffic pattern, specify that the outer S-VLAN tag is popped on incoming packets. To accommodate a scenario in which the traffic flow is reversed, and the VTEP functions as an egress VTEP that receives a single-tagged packet from C-VLAN 200, you can optionally specify that an outer S-VLAN tag is added, or pushed, on outgoing packets.

```
[edit interfaces]
user@switch# set xe-0/0/0 unit 100 vlan-id 100
user@switch# set xe-0/0/0 unit 100 input-vlan-map pop
user@switch# set xe-0/0/0 unit 100 output-vlan-map push
```

4. Create a VLAN named vlan_1, and map it to logical interface xe-0/0/0.100 and VNI 1001. Also specify that the logical interface retains the inner C-VLAN tag while encapsulating a packet and the support of ingress node replication if the interface is multihomed.

```
[edit vlans]
user@switch# set vlan_1 interface xe-0/0/0.100
user@switch# set vlan_1 vxlan vni 1001
user@switch# set vlan_1 vxlan encapsulate-inner-vlan
user@switch# set vlan_1 vxlan ingress-node-replication
```

Egress VTEP Configuration for Traffic Pattern 1

CLI Quick Configuration

To quickly configure the egress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

NOTE: You do not need to issue the **set protocols l2-learning decapsulate-accept-inner-vlan** command on the QFX10000 line of switches because, by default, these switches retain the inner C-VLAN tag while a packet is tunneled.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set vlans vlan_1 interface xe-0/0/0.100
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
set vlans vlan_1 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure the egress VTEP for traffic pattern 1:

1. On all Juniper Networks devices except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100. Also, configure logical interface 100, and associate it with VLAN 100.

```
[edit interfaces]
user@switch# set xe-0/0/0 flexible-vlan-tagging
user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
user@switch# set xe-0/0/0 unit 100 vlan-id 100
```

3. Create a VLAN named vlan_1, and map it to logical interface xe-0/0/0.100 and VNI 1001. Also specify that the logical interface retains the inner C-VLAN tag while encapsulating a packet and the support of ingress node replication if the interface is multihomed.

```
[edit vlans]
user@switch# set vlan_1 interface xe-0/0/0.100
user@switch# set vlan_1 vxlan vni 1001
user@switch# set vlan_1 vxlan encapsulate-inner-vlan
user@switch# set vlan_1 vxlan ingress-node-replication
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

Configuring Traffic Pattern 2: Mapping a Range of C-VLANs to an S-VLAN, and Pushing an S-VLAN Tag

Introduction

For this traffic pattern, the ingress VTEP in an EVPN-VXLAN overlay network receives a packet tagged with a C-VLAN ID, one of which is included in a configured range of C-VLAN IDs that are mapped to a particular S-VLAN. After the packet is tunneled over the Layer 3 network, the egress VTEP retains the C-VLAN tag and pushes an outer tag for that particular S-VLAN on the packet.

NOTE: The QFX10000 line of switches support this traffic pattern on aggregated Ethernet and non-aggregated Ethernet interfaces. The remaining QFX Series and EX4600 switches support this traffic pattern only on non-aggregated Ethernet interfaces.

NOTE: QFX Series and EX4600 switches do not support the pop and push actions with a configured range of VLANs.

Ingress VTEP Configuration for Traffic Pattern 2

NOTE: This configuration focuses on traffic pattern 2 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a two-layer IP fabric, see "Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center" on page 302.

CLI Quick Configuration

To quickly configure the ingress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

NOTE: You do not need to issue the **set protocols l2-learning decapsulate-accept-inner-vlan** command on the QFX10000 line of switches because, by default, these switches retain the inner C-VLAN tag while a packet is tunneled.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/5 flexible-vlan-tagging
set interfaces xe-0/0/5 encapsulation extended-vlan-bridge
set interfaces xe-0/0/5 unit 100 vlan-id-list 100-200
set vlans vlan_range1 interface xe-0/0/5.100
set vlans vlan_range1 vxlan vni 1001
set vlans vlan_range1 vxlan encapsulate-inner-vlan
set vlans vlan_range1 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure the ingress VTEP for traffic pattern 2:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

```
[edit protocols l2-learning]
user@switch# set decapsulate-accept-inner-vlan
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100. Also, for the physical interface, configure logical interface 100 and map it to C-VLANs 100 through 200.

```
[edit interfaces]
user@switch# set xe-0/0/5 flexible-vlan-tagging
user@switch# set xe-0/0/5 encapsulation extended-vlan-bridge
user@switch# set xe-0/0/5 unit 100 vlan-id-list 100-200
```

3. Create a VLAN named vlan_range1, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet and the support of ingress node replication if the interface is multihomed.

```
[edit vlans]
user@switch# set vlan_range1 interface xe-0/0/5.100
user@switch# set vlan_range1 vxlan vni 1001
user@switch# set vlan_range1 vxlan encapsulate-inner-vlan
user@switch# set vlan_range1 vxlan ingress-node-replication
```

Egress VTEP Configuration for Traffic Pattern 2

CLI Quick Configuration

To quickly configure the egress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

NOTE: You do not need to issue the **set protocols l2-learning decapsulate-accept-inner-vlan** command on the QFX10000 line of switches because, by default, these switches retain the inner C-VLAN tag while a packet is tunneled.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set interfaces xe-0/0/0 unit 100 input-vlan-map pop
set interfaces xe-0/0/0 unit 100 output-vlan-map push
set vlans v100 interface xe-0/0/0.100
```

```
set vlans v100 vxlan vni 1001
set vlans v100 vxlan encapsulate-inner-vlan
set vlans v100 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure the egress VTEP for traffic pattern 2:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

   ```
   [edit protocols l2-learning]
   user@switch# set decapsulate-accept-inner-vlan
   ```

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/0 flexible-vlan-tagging
   user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
   ```

3. Create logical interface 100, and associate it with S-VLAN 100. Also, specify that when logical interface 100 receives a packet without an outer S-VLAN tag, the interface pushes outer S-VLAN tag 100 on the outgoing packet.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/0 unit 100 vlan-id 100
   user@switch# set xe-0/0/0 unit 100 input-vlan-map pop
   user@switch# set xe-0/0/0 unit 100 output-vlan-map push
   ```

   NOTE: If you include the **push** configuration statement at the **[edit interfaces unit output-vlan-map]** hierarchy level, you must also include the **pop** configuration statement at the **[edit interfaces unit input-vlan-map]** hierarchy level to prevent an error when committing the configuration.

4. Create a VLAN named v100, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet and the support of ingress node replication if the interface is multihomed.

   ```
   [edit vlans]
   user@switch# set v100 interface xe-0/0/0.100
   user@switch# set v100 vxlan vni 1001
   user@switch# set v100 vxlan encapsulate-inner-vlan
   user@switch# set v100 vxlan ingress-node-replication
   ```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

Configuring Traffic Pattern 3: Retaining S-VLAN and C-VLAN Tags

Introduction

For this traffic pattern, the ingress and egress VTEPs in an EVPN-VXLAN overlay network must handle Q-in-Q data packets that are single- or double-tagged. For both single- and double-tagged packets, the ingress and egress VTEPs encapsulate and de-encapsulate the packets without making any changes to the tag(s).

NOTE: QFX Series and EX4600 switches support this traffic pattern on both aggregated Ethernet and non-aggregated Ethernet interfaces.

Ingress and Egress VTEP Configuration for Traffic Pattern 3

NOTE: This configuration focuses on traffic pattern 3 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a two-layer IP fabric, see "Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center" on page 302.

CLI Quick Configuration

To quickly configure the ingress and egress VTEPs, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

NOTE: You do not need to issue the **set protocols l2-learning decapsulate-accept-inner-vlan** command on the QFX10000 line of switches because, by default, these switches retain the inner C-VLAN tag while a packet is tunneled.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/15 flexible-vlan-tagging
set interfaces xe-0/0/15 encapsulation extended-vlan-bridge
set interfaces xe-0/0/15 unit 100 vlan-id 100
set interfaces xe-0/0/15 unit 201 vlan-id 201
set vlans vlan_100 interface xe-0/0/15.100
set vlans vlan_100 vxlan vni 1001
```

```
set vlans vlan_100 vxlan encapsulate-inner-vlan
set vlans vlan_100 vxlan ingress-node-replication
set vlans vlan_201 interface xe-0/0/15.201
set vlans vlan_201 vxlan vni 1002
set vlans vlan_201 vxlan encapsulate-inner-vlan
set vlans vlan_201 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure the ingress and egress VTEP for traffic pattern 3:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEPs to retain the inner VLAN tag while de-encapsulating a packet.

   ```
   [edit protocols l2-learning]
   user@switch# set decapsulate-accept-inner-vlan
   ```

   NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single- and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100. Also, on the physical interface, create logical interfaces 100 and 201, and associate them with S-VLAN 100 and C-VLAN 201, respectively.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/15 flexible-vlan-tagging
   user@switch# set xe-0/0/15 encapsulation extended-vlan-bridge
   user@switch# set xe-0/0/15 unit 100 vlan-id 100
   user@switch# set xe-0/0/15 unit 201 vlan-id 201
   ```

3. Create a VLAN named vlan_100, and map it to logical interface 100 and VNI 1001. Also create a VLAN named vlan_201, and map it to logical interface 201 and VNI 1002. Also specify that the logical interfaces retain the inner VLAN tag while encapsulating a packet and the support of ingress node replication if the interfaces are multihomed.

   ```
   [edit vlans]
   user@switch# set vlan_100 interface xe-0/0/15.100
   user@switch# set vlan_100 vxlan vni 1001
   user@switch# set vlan_100 vxlan encapsulate-inner-vlan
   user@switch# set vlan_100 vxlan ingress-node-replication
   user@switch# set vlan_201 interface xe-0/0/15.201
   user@switch# set vlan_201 vxlan vni 1002
   user@switch# set vlan_201 vxlan encapsulate-inner-vlan
   user@switch# set vlan_201 vxlan ingress-node-replication
   ```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

Configuring Traffic Pattern 4: Popping and Later Pushing an S-VLAN Tag

Introduction

For this traffic pattern, the ingress and egress VTEPs in an EVPN-VXLAN overlay network must handle double-tagged Q-in-Q traffic. The ingress VTEP retains the inner C-VLAN tag and removes, or pops, the outer S-VLAN tag. After the packets are tunneled over the Layer 3 network, the egress VTEP pushes the S-VLAN tag back on the packet.

NOTE: QFX Series and EX4600 switches support this traffic patterns on both aggregated Ethernet and non-aggregated Ethernet interfaces.

NOTE: This configuration focuses on traffic pattern 4 only. It does not provide the configuration for EVPN and all aspects of VXLAN. For a more comprehensive EVPN-VXLAN configuration for a two-layer IP fabric, see "Example: Configuring IRB Interfaces in an EVPN-VXLAN Environment to Provide Layer 3 Connectivity for Hosts in a Data Center" on page 302.

Configuration for Ingress VTEP for Traffic Pattern 4

CLI Quick Configuration

To quickly configure the ingress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

NOTE: You do not need to issue the **set protocols l2-learning decapsulate-accept-inner-vlan** command on the QFX10000 line of switches because, by default, these switches retain the inner C-VLAN tag while a packet is tunneled.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation extended-vlan-bridge
set interfaces xe-0/0/0 unit 100 vlan-id 100
set interfaces xe-0/0/0 unit 100 input-vlan-map pop
set interfaces xe-0/0/0 unit 100 output-vlan-map push
set vlans vlan_1 interface xe-0/0/0.100
```

```
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
set vlans vlan_1 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure the ingress VTEP for traffic pattern 4:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

   ```
   [edit protocols l2-learning]
   user@switch# set decapsulate-accept-inner-vlan
   ```

   NOTE: To support the VXLAN tunneling of Q-in-Q packets, you must configure both ingress and egress VTEP to retain the inner C-VLAN tag while de-encapsulating a packet.

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/0 flexible-vlan-tagging
   user@switch# set xe-0/0/0 encapsulation extended-vlan-bridge
   ```

3. On physical interface xe-0/0/0, create logical interface 100, and associate it with S-VLAN 100. Also, assuming that the ingress VTEP receives a double-tagged packet as described in this traffic pattern, specify that the outer S-VLAN tag is popped on incoming packets. To accommodate a scenario in which the traffic flow is reversed, and the VTEP functions as an egress VTEP that receives a single-tagged packet from C-VLAN 200, you can optionally specify that an outer S-VLAN tag is added, or pushed, on outgoing packets.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/0 unit 100 vlan-id 100
   user@switch# set xe-0/0/0 unit 100 input-vlan-map pop
   user@switch# set xe-0/0/0 unit 100 output-vlan-map push
   ```

4. Create a VLAN named vlan_1, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet and the support of ingress node replication if the interface is multihomed.

   ```
   [edit vlans]
   user@switch# set vlan_1 interface xe-0/0/0.100
   user@switch# set vlan_1 vxlan vni 1001
   user@switch# set vlan_1 vxlan encapsulate-inner-vlan
   user@switch# set vlan_1 vxlan ingress-node-replication
   ```

Configuration for Egress VTEP for Traffic Pattern 4

CLI Quick Configuration

To quickly configure the egress VTEP, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

NOTE: You do not need to issue the **set protocols l2-learning decapsulate-accept-inner-vlan** command on the QFX10000 line of switches because, by default, these switches retain the inner C-VLAN tag while a packet is tunneled.

```
set protocols l2-learning decapsulate-accept-inner-vlan
set interfaces xe-0/0/5 flexible-vlan-tagging
set interfaces xe-0/0/5 encapsulation extended-vlan-bridge
set interfaces xe-0/0/5 unit 100 vlan-id 100
set interfaces xe-0/0/5 unit 100 input-vlan-map pop
set interfaces xe-0/0/5 unit 100 output-vlan-map push
set vlans vlan_1 interface xe-0/0/5.100
set vlans vlan_1 vxlan vni 1001
set vlans vlan_1 vxlan encapsulate-inner-vlan
set vlans vlan_1 vxlan ingress-node-replication
```

Step-by-Step Procedure

To configure the egress VTEP for traffic pattern 4:

1. On all supported Juniper Networks switches except the QFX10000 line of switches, configure the VTEP to retain the inner VLAN tag while de-encapsulating a packet.

   ```
   [edit protocols l2-learning]
   user@switch# set decapsulate-accept-inner-vlan
   ```

2. Configure the physical interface to support the simultaneous transmission of 802.1Q VLAN single-tagged and double-tagged packets on its logical interfaces and to accept packets carrying TPID 0x8100.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/5 flexible-vlan-tagging
   user@switch# set xe-0/0/5 encapsulation extended-vlan-bridge
   ```

3. Create logical interface 100, and associate it with S-VLAN 100. Also, specify that when logical interface 100 receives a packet without an outer S-VLAN tag, the interface pushes outer S-VLAN tag 100 on the outgoing packet.

   ```
   [edit interfaces]
   user@switch# set xe-0/0/5 unit 100 vlan-id-list 100
   user@switch# set xe-0/0/5 unit 100 input-vlan-map pop
   ```

```
user@switch# set xe-0/0/5 unit 100 output-vlan-map push
```

NOTE: If you include the **push** configuration statement at the **[edit interfaces unit output-vlan-map]** hierarchy level, you must also include the **pop** configuration statement at the **[edit interfaces unit input-vlan-map]** hierarchy level to prevent an error when committing the configuration.

4. Create a VLAN named vlan_1, and map it to logical interface 100 and VNI 1001. Also specify that the logical interface retains the inner VLAN tag while encapsulating a packet and the support of ingress node replication if the interface is multihomed.

```
[edit vlans]
user@switch# set vlan_1 interface xe-0/0/5.100
user@switch# set vlan_1 vxlan vni 1001
user@switch# set vlan_1 vxlan encapsulate-inner-vlan
user@switch# set vlan_1 vxlan ingress-node-replication
```

NOTE: To support the tunneling of Q-in-Q packets, you must configure both ingress and egress VTEPs to retain the inner C-VLAN tag while encapsulating a packet.

PART 3

EVPN-MPLS

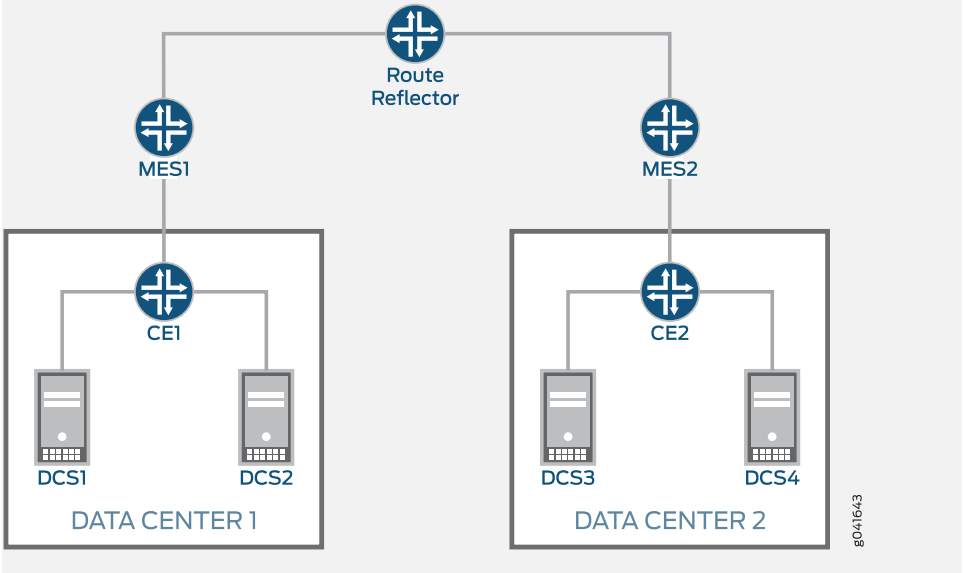
CHAPTER 16

Overview

EVPN Overview

An Ethernet VPN (EVPN) enables you to connect dispersed customer sites using a Layer 2 virtual bridge. As with other types of VPNs, an EVPN consists of customer edge (CE) devices (host, router, or switch) connected to provider edge (PE) routers. The PE routers can include an MPLS edge switch (MES) that acts at the edge of the MPLS infrastructure. Either an MX Series 5G Universal Routing Platform or a standalone switch can be configured to act as an MES. You can deploy multiple EVPNs within a service provider network, each providing network connectivity to a customer while ensuring that the traffic sharing on that network remains private. Figure 51 on page 550 illustrates a typical EVPN deployment. Traffic from Data Center 1 is transported over the service provider's network through MES1 to MES2 and then onto Data Center 2. DCS1, DCS2, DCS3, and DCS4 are the data center switches.

Figure 51: EVPN Connecting Data Center 1 and Data Center 2



The MESs are interconnected within the service provider's network using label-switched paths (LSPs). The MPLS infrastructure allows you to take advantage of the MPLS functionality provided by the Junos OS, including fast reroute, node and link protection, and standby secondary paths. For EVPNs, learning between MESs takes place in the control plane rather than in the data plane (as is the case with traditional network bridging). The control plane provides greater control over the learning process, allowing you to restrict which devices discover information about the network. You can also apply policies on the MESs, allowing you to carefully control how network information is distributed and processed. EVPNs utilize the BGP control plane infrastructure, providing greater scale and the ability to isolate groups of devices (hosts, servers, virtual machines, and so on) from each other.

The MESs attach an MPLS label to each MAC address learned from the CE devices. This label and MAC address combination is advertised to the other MESs in the control plane. Control plane learning enables load balancing and improves convergence times in the event of certain types of network failures. The learning process between the MESs and the CE devices is completed using the method best suited to each CE device (data plane learning, IEEE 802.1, LLDP, 802.1aq, and so on).

The policy attributes of an EVPN are similar to an IP VPN (for example, Layer 3 VPNs). Each EVPN routing instance requires that you configure a route distinguisher (RD) and one or more route targets (RTs). A CE device attaches to an EVPN routing instance on an MES through an Ethernet interface that might be configured for one or more VLANs.

The following features are available for EVPNs:

- Ethernet connectivity between data centers spanning metropolitan area networks (MANs) and WANs
- One VLAN for each MAC VPN

- Automatic RDs
- Dual-homed EVPN connection with active/standby multihoming
- The following Juniper Networks devices support active/active multihoming:
  - Starting in Junos OS Releases 14.2R6 and 16.1R1, MX Series routers. It is not supported in Junos OS Release 15.1.
  - Starting in Junos OS Releases 16.1R4 and 16.2R2, EX9200 switches.
- Ethernet VPN (EVPN) support, including EVPN-MPLS, EVPN + VXLAN, and PBB EVPN, has been extended to logical systems running only on MX devices. The same EVPN options and performance that are available in the default EVPN instance are available in a logical system. Configure EVPN on a logical system under the **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols evpn]** hierarchy.

The following feature is not supported for EVPNs:

- Graceful restart, graceful Routing Engine switchover (GRES), and nonstop active routing (NSR) is not supported in releases prior to Junos OS Release 16.1.

EVPN MPLS Features Supported by QFX10000 Switches

Starting in Junos OS 17.4R1, QFX10000 switches support EVPN with MPLS as its data plane, as defined in RFC 7432. The following features are supported:

- Layer 2 VLANs using the default-switch routing instance. An EVPN instance (EVI) is not supported.
- EVPN MPLS mulithoming active-active support
- VLAN-aware bundle service interface without translation (support is limited to 4K VLANs as only the default-switching instance is supported)
- Ingress multicast replication
- Mac mobility

Release History Table

Release	Description
17.4	Ethernet VPN (EVPN) support, including EVPN-MPLS, EVPN + VXLAN, and PBB EVPN, has been extended to logical systems running only on MX devices. The same EVPN options and performance that are available in the default EVPN instance are available in a logical system. Configure EVPN on a logical system under the **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols evpn]** hierarchy.

Related Documentation

- Supported EVPN Standards on page 553
- EVPN Multihoming Overview on page 29
- Overview of MAC Mobility on page 8

EVPN Overview for Switches

An Ethernet VPN (EVPN) enables you to connect a group of dispersed customer sites using a Layer 2 virtual bridge. As with other types of VPNs, an EVPN comprises customer edge (CE) devices (host, router, or switch) connected to provider edge (PE) devices. The PE devices can include an MPLS edge switch (MES) that acts at the edge of the MPLS infrastructure. For the initial deployment of EVPNs using Juniper Networks equipment, you can configure an EX9200 switch to act as an MES. You can deploy multiple EVPNs within the network, each providing network connectivity to customers while ensuring that traffic sharing that network remains private.

The MESs are interconnected within the network by using label-switched paths (LSPs). The MPLS infrastructure allows you to take advantage of the MPLS functionality provided by the Junos operating system (Junos OS), including fast reroute, node and link protection, and standby secondary paths. For EVPNs, learning between MESs takes place in the control plane rather than in the data plane (as is the case with traditional network bridging). The control plane provides greater control over the learning process, allowing you to restrict which devices discover information about the network. You can also apply policies on the MESs, allowing you to carefully control how network information is distributed and processed. EVPNs utilize the BGP control plane infrastructure, providing greater scale and the ability to isolate groups of devices (hosts, servers, virtual machines, and so on) from each other.

The MESs attach an MPLS label to each MAC address learned from the CE devices. This label and MAC address combination is advertised to the other MESs in the control plane. Control plane learning enables load balancing and improves convergence times in the event of certain types of network failures. The learning process between the MESs and the CE devices is completed using the method best suited to each CE device (data plane learning, IEEE 802.1, LLDP, 802.1aq, and so on).

The policy attributes of an EVPN are similar to an IP VPN (for example, Layer 3 VPNs). Each EVPN routing instance requires that you configure a route distinguisher and one or more route targets. A CE device attaches to an EVPN routing instance on an MES through an Ethernet interface that might be configured for one or more VLANs.

The following features are available for EVPNs:

- Ethernet connectivity between data centers spanning metropolitan area networks (MANs) and WANs
- One or more VLANs for each MAC VPN
- Automatic route distinguishers
- Dual-homed EVPN connection with active standby multihoming
- Starting with Junos OS Releases 16.1R4 and 16.2R2, the active-active mode for EVPN multihoming is supported.
- Starting with Junos OS Relese 17.3R1, both pure type-5 routes and standard type-5 routes are supported on EX9200 switches. Use this feature, which advertises IP prefixes through EVPN, when the Layer 2 domain does not exist at the remote data centers or

metro network peering points. For more information about how to configure, see ip-prefix-routes.

- Starting with Junos OS OS Release 17.3R1, VXLAN encapsulation is supported. Previously, only MPLS encapsulation is supported.

The following features are not supported for EVPNs:

- Graceful restart, graceful Routing Engine switchover (GRES), and nonstop active routing (NSR)

Release History Table

Release	Description
17.3R1	Starting with Junos OS Relese 17.3R1, both pure type-5 routes and standard type-5 routes are supported on EX9200 switches.
17.3R1	Starting with Junos OS OS Release 17.3R1, VXLAN encapsulation is supported. Previously, only MPLS encapsulation is supported.
16.1R4	Starting with Junos OS Releases 16.1R4 and 16.2R2, the active-active mode for EVPN multihoming is supported.

Related Documentation

- Supported EVPN Standards on page 553
- Example: Configuring EVPN Active-Active Multihoming on page 123

Supported EVPN Standards

Junos OS supports the following RFCs and Internet drafts that define standards for EVPNs:

- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4761, *Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*
- RFC 7432, *BGP MPLS-Based Ethernet VPN*

  The following features are not supported:

  - Automatic derivation of Ethernet segment (ES) values. Only static ES configurations are supported.
  - Host proxy ARP.
- Internet draft draft-ietf-spring-segment-routing-13, *Segment Routing Architecture*
- Internet draft draft-ietf-spring-segment-routing-mpls-11, *Segment Routing with MPLS data plane*
- Internet draft draft-ietf-isis-segment-routing-extensions-13, *IS-IS Extensions for Segment Routing*

Related Documentation

- EVPN Overview on page 549
- *Accessing Standards Documents on the Internet*

Migrating from FEC128 LDP-VPLS to EVPN Overview

For service providers with virtual private LAN service (VPLS) networks and Ethernet VPN (EVPN) networks, there is a need to interconnect these networks. Prior to Junos OS Release 17.3, a logical tunnel interface on the interconnection point of the VPLS and EVPN routing instances was used for this purpose. In this case, the provider edge (PE) devices in each network were unaware of the PE devices in the other technology network. Starting in Junos OS Release 17.3, a solution is introduced for enabling staged migration from FEC128 LDP-VPLS toward EVPN on a site-by-site basis for every VPN routing instance. In this solution, the PE devices running EVPN and VPLS for the same VPN routing instance and single-homed segments can coexist. During migration, there is minimal impact to the customer edge (CE) device-to-CE device traffic forwarding for affected customers.

The following sections describe the migration from LDP-VPLS to EVPN:

- Technology Overview and Benefits on page 554
- FEC128 LDP-VPLS to EVPN Migration on page 555
- Sample Configuration for LDP-VPLS to EVPN Migration on page 556
- Reverting to VPLS on page 559
- LDP-VPLS to EVPN Migration and Other Features on page 560

Technology Overview and Benefits

Virtual private LAN service (VPLS) is an Ethernet-based point-to-multipoint Layer 2 VPN. This technology allows you to connect geographically dispersed data center LANs to each other across an MPLS backbone while maintaining Layer 2 connectivity. The high availability features defined in VPLS standards (such as LER dual homing) and topology autodiscovery features using BGP signaling make VPLS scalable and easy to deploy. Because VPLS uses MPLS as its core, it provides low latency variation and statistically bound low convergence times within the MPLS network.

Ethernet VPN (EVPN), on the other hand, is a combined Layer 2 and Layer 3 VPN solution that is more scalable, resilient, and efficient than current technologies. It provides several benefits including greater network efficiency, reliability, scalability, virtual machine (VM) mobility, and policy control for service providers and enterprises.

Although VPLS is a widely deployed Layer 2 VPN technology, service provider networks migrate to EVPN because of the scaling benefits and ease of deployment. Some of the benefits of EVPN include:

- Control plane traffic is distributed with BGP and the broadcast and multicast traffic is sent using a shared multicast tree or with ingress replication.
- Control plane learning is used for MAC and IP addresses instead of data plane learning. MAC address learning requires the flooding of unknown unicast and ARP frames, whereas IP address learning does not require any flooding.
- Route reflector is used to reduce a full mesh of BGP sessions among PE devices to a single BGP session between a PE device and the route reflector.
- Autodiscovery with BGP is used to discover PE devices participating in a given VPN, PE devices participating in a given redundancy group, tunnel encapsulation types, multicast tunnel type, and multicast members.
- All-active multihoming is used. This allows a given CE device to have multiple links to multiple PE devices, and traffic traversing to-and-from that CE device fully utilizes all of these links (Ethernet segment).
- When a link between a CE device and a PE device fails, the PE devices for that EVPN instance (EVI) are notified of the failure with the withdrawal of a single EVPN route. This allows those PE devies to remove the withdrawing PE device as a next hop for every MAC address associated with the failed link (mass withdrawal).

FEC128 LDP-VPLS to EVPN Migration

Some service providers want to preserve their investments in VPLS. This leads to the need to connect the old VPLS networks to new networks that run EVPN. For this purpose, logical tunnel interfaces on the interconnection point of the VPLS and EVPN routing instances were used. However, all the other PE devices belonged either to the VPLS network or to the EVPN network and were unaware of the other technology.

Starting in Junos OS Release 17.3, EVPN can be introduced into an existing VPLS network in a staged manner, with minimal impact to VPLS services. On a VPLS PE device, some customers can be moved to EVPN, while other customers continue to use VPLS pseudowires. Other PE devices can be entirely VPLS and switching customers on other PE devices to EVPN. This solution provides support for the seamless migration Internet draft (expires January ,2018), *(PBB-)EVPN Seamless Integration with (PBB-)VPLS*.

The seamless migration from FEC128 LDP-VPLS to EVPN solution supports the following functionality:

- Allow for staged migration toward EVPN on a site-by-site basis per VPN instance. For instance, new EVPN sites to be provisioned on EVPN PE devices.
- Allow for the coexistence of PE devices running both EVPN and VPLS for the same VPN instance and single-homed segments.

In the LDP-VPLS to EVPN migration, the PE device where some customers have been migrated to EVPN while other customers are being served using VPLS is called a super

PE device. As super PE devices discover other super PE devices within a routing instance, they use EVPN forwarding to communicate with other super PE devices and VPLS pseudowires to PE devices running VPLS. The PE device with no EVPN awareness, and running only VPLS for all the customers, is called a VPLS PE device.

The CE device connected to a super PE can reach CE devices connected to EVPN-only PE devices or VPLS-only PE devices, but CE devices connected to EVPN-only PE devices cannot reach CE devices connected to VPLS-only PE devices.

Because the migration from LDP-VPLS to EVPN is supported on a per-routing instance basis, and if the routing instance is serving multiple customers on a PE device, all are migrated together. EVPN is responsible for setting up data forwarding between the PE devices upgraded to EVPN, while VPLS continues to set up data forwarding to PE devices that run VPLS. There should be zero impact for customers that still use VPLS pseudowire on all the PE devices.

NOTE:

The following features are not supported with the LDP-VPLS to EVPN migration:

- Migration from FEC129 VPLS to EVPN.
- Migration from BGP-VPLS to EVPN.
- Migration of VPLS virtual switch to EVPN virtual switch.
- Migration of VPLS routing instance to EVPN virtual switch.
- Migration of VPLS routing instance or PBB-VPLS to PBB-EVPN.
- Seamless migration from EVPN back to VPLS.
- Enhancing EVPN to support the set of tools or statements and commands that VPLS supports.
- Active-active and active-standby multihoming. The migration to EVPN is supported only on single-homed deployments.
- Spanning all-active across EVPN and VPLS PE devices does not work, because the all-active multihoming feature is not supported on VPLS.
- Connecting EVPN-only PE devices with VPLS-only PE devices through super PE devices.
- IPv6, logical systems, multichassis support, and SNMP, because they are currently not supported on EVPN.

Sample Configuration for LDP-VPLS to EVPN Migration

The following sections provide the sample configuration required for performing the LDP-VPLS to EVPN migration.

- LDP-VPLS Configuration on page 557
- EVPN Migration Configuration on page 558

LDP-VPLS Configuration

A typical static LDP-VPLS routing instance configuration is as follows:

```
user@host# show routing-instance foo
instance-type vpls;
vlan-id 100; (not needed for VLAN bundle service)
interface ge-2/0/0.590;
interface ae500.590;
routing-interface irb.0;
forwarding-options {
   family vpls {
      filter {
         input UNKNOWN-UNICAST;
      }
   }
}
protocols {
   vpls {
      control-word;
      encapsulation-type ethernet-vlan;
      enable-mac-move-action;
      mac-table-size {
         100000;
         packet-action drop;
      }
      mac-table-aging-time ;
      interface-mac-limit {
         100000;
         packet-action drop;
      }
      no-tunnel-services; (use label-switched interfaces)
      vpls-id 245015;
      mtu 1552;
      ignore-mtu-mismatch;
      mac-flush {
         any-spoke;
      }
      no-vlan-id-validate;
      neighbor 192.168.252.64 {
         psn-tunnel-endpoint 10.0.0.31;
         pseudowire-status-tlv;
         revert-time 60;
         backup-neighbor 192.168.252.65 {
            psn-tunnel-endpoint 10.0.0.32;
            hot-standby;
         }
      }
      mesh-group Spoke { (access label-switched interface toward spoke)
      local-switching;
      neighbor 192.168.252.66 {
         psn-tunnel-endpoint 10.0.0.41;
         pseudowire-status-tlv;
      }
```

```
    neighbor 192.168.252.67 {
      psn-tunnel-endpoint 10.0.0.42;
      pseudowire-status-tlv;
    }
  }
  connectivity-type permanent;
}
```

```
user@host# show interfaces ge-2/0/0.590
encapsulation vlan-vpls;
vlan-id 590;
output-vlan-map {
  swap;
  tag-protocol-id 0x8100;
  inner-vlan-id 590;
}
family vpls {
  filter {
    input-list [ listA ];
    output-list listB;
  }
}
```

EVPN Migration Configuration

To perform the FEC128 LDP-VPLS to EVPN migration, do the following:

1. On the backup Routing Engine, load Junos OS Release 17.3R1.

2. Perform in-service software upgrade (ISSU) to acquire mastership. Ensure that the VPLS unified ISSU does not have any impact on the VPLS forwarding.

3. Identify routing instances (customers) that need to be migrated to EVPN.

4. Enable EVPN in a single routing instance.

   - Change routing instance type to **evpn**, and include the **evpn** statement at the **[edit routing-instances *routing-intance-name* protocols]** hierarchy level, and also include the **vpls** statement at the same hierarchy to support VPLS commands.

     For example:

     ```
     [edit routing-instances routing-instance-name]
     instance-type evpn;
     interface ge-2/0/0.590;
     interface ae500.590;
     routing-interface irb.0;
     route-distinguisher 1.1.1.1:50; (add for LDP-VPLS)
     vrf-target target:100:100; (add for LDP-VPLS)
     forwarding-options {
       family vpls {
         filter {
     ```

```
        input UNKNOWN-UNICAST;
      }
    }
  }
  protocols {
    vpls { (supports all existing VPLS commands)
  }
```

5. Enable family EVPN signaling in BGP.

   For example:

```
protocols {
  bgp {
    local-as 102;
    group 2mx {
      type internal;
      local-address 81.1.1.1;
      family evpn {
      signaling;
    }
    neighbor 81.2.2.2;
    neighbor 81.9.9.9;
  }
}
```

After the configuration for the EVPN migration is committed, the routing protocol process and the Layer 2 address learning process start building the EVPN state to reflect interfaces, bridge domains, peers and routes. The locally learnt MAC addresses are synchronized by the Layer 2 address learning process in the instance.vpls.0 to the routing protocol process. When a local MAC ages out in the instance.vpls.0, the routing protocol process is informed by the Layer 2 address learning process.

When an EVPN peer is learnt, the routing protocol process sends a new message to the Layer 2 address learning process to remove the peer's label-switched interface or virtual tunnel logical interface from the VE mesh group and disables MAC-learning on it. The EVPN IM next-hop is then added to the VE mesh group. The EVPN behavior in the routing protocol process of learning MAC addresses over BGP and informing Layer 2 address learning process of the MPLS next hop is maintained.

The VPLS statements and commands continue to apply to the VPLS pseudowires between the PE devices and the MAC addresses learnt over them. The EVPN statements and commands apply to PE devices running EVPN.

Reverting to VPLS

If the EVPN migration runs into issues, you can revert back to VPLS until the issue is understood. The routing instance is reverted from a super PE to a VPLS PE in a non-catastrophic manner by enabling the following configuration:

```
[edit routing-instances routing-instance-name]
user@host# set instance-type vpls
```

```
user@host# delete protocols evpn
user@host# delete route-distinguisher (if running LDP-VPLS)
user@host# delete vrf-target (if running LDP-VPLS)
```

On reverting the EVPN migration to VPLS, the following happens:

1. The EVPN state information is deleted.
2. There is a trigger for withdrawal of EVPN control plane routes.
3. The routing protocol process sends a new message to the Layer 2 address learning process with the label-switched interface or the virtual tunnel logical interface for the routing instance and peer.
4. The label-switched or virtual tunnel interface adds the new message to the flood group and MAC learning is enabled.
5. The egress IM next hop is deleted by the routing protocols process, prompting the Layer 2 address learning process to remove it from the flood group.
6. Remote MAC addresses are learned again over the label-switched interface or virtual tunnel logical interface.

LDP-VPLS to EVPN Migration and Other Features

Table 4 on page 26 describes the functionality of some of the related features, such as multihoming and integrated routing and bridging (IRB) with the LDP-VPLS to EVPN migration.

Table 18: EVPN Migration and Other Features Support

Feature	Supported Functionality in EVPN Migration
MAC move	MAC moves are supported between VPLS-only PE device and super PE devices. When a MAC address moves from a VPLS-only PE device to a super PE device, it is learned over BGP, and the routing protocol process informs the Layer 2 address learning process of the EVPN next hop to be updated in the foo.vpls.0 routing table. When a MAC address moves from a super PE device to a VPLS-only PE device, it is learned in the Packet Forwarding Engine on the label-switched interface or virtual tunnel interface. The Layer 3 address learning process updates it to VPLS or the label-switched interface next hop. When the type 2 route is withdrawn by EVPN BGP, the MAC address is not deleted from the forwarding table, so there is no loss of data. The forwarding MAC table is shared by VPLS and EVPN. Some attributes, such as **mac-table-size** and **mac-table-aging-time** could be configured under both EVPN and VPLS. When there is a conflict, the values under EVPN take precedence.

Table 18: EVPN Migration and Other Features Support (continued)

Feature	Supported Functionality in EVPN Migration
IRB	No changes needed in IRB. On a super PE device, EVPN populates the /32 host routes learned over MAC+IP type 2 routes from EVPN peers in a Layer 3 virtual routing and forwarding, while VPLS IRB forwarding using subnet routes works on sites still running VPLS.
Hierarchical VPLS	In an H-VPLS network with hub-and-spoke PE devices, when the hub PE device is migrated to EVPN, local MAC addresses learned over the access label-switched or virtual tunnel interface need to be advertised to BGP, so that the other EVPN-only PE devices or super PE devices can reach them. Take the following into consideration when migrating an H-VPLS network to EVPN: • Hubs typically have local switching enabled as interspoke traffic is forwarded through the hub. If spokes alone are migrated to EVPN and spokes have Layer 3 or MPLS reachability to each other, the label-switched or virtual tunnel interface to the hub and EVPN next hop (remote spoke) is present in the VPLS edge (VE) floodgroup. This results in two copies of broadcast, unknown unicast, and multicast (BUM) traffic received by the remote spoke. One option to avoid this behavior is to migrate the hubs to EVPN too. • EVPN is not aware of hierarchy. All peers are considered core-facing. Once hubs and spokes are migrated to EVPN, split horizon prevents the BUM traffic from being forwarded to other core-facing PE devices.
ESI configuration	Ethernet segment identifier (ESI) is configured at the physical interface or port level.

Related Documentation

- EVPN Overview on page 549

CHAPTER 17

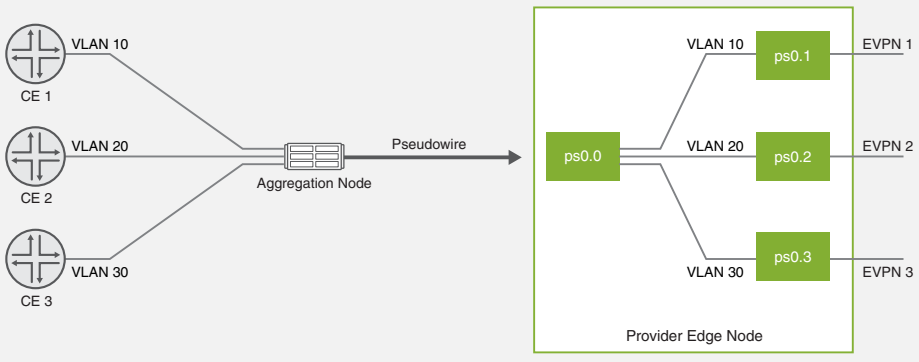
Pseudowire Termination at an EVPN

Overview of Pseudowire Termination at an EVPN

Pseudowires provide point-to-point service over an MPLS or Layer 2 circuit. They enable providers to extend their Layer 2 networks. A pseudowire tunnel terminates on a logical interface on the PE router with the transport logical interface configured for interoperability with the access or aggregation device on the other end of the pseudowire tunnel. It is identified as psn.0 and supports both port-based and VLAN-based encapsulation over pseudowire. The corresponding service logical interfaces are identified as psn.1 to psn.n and are configured to support EVPN in the EVPN routing instance.

Figure 52 on page 563 shows a port-based pseudowire tunnel that originates on an aggregation node carrying VLAN traffic from several CE devices. The pseudowire terminates in a transport logical interface (ps0.0) on a single-homed PE router, where the VLAN traffic is demultiplexed into different service logical interfaces(ps0.1, ps0.2, and ps0.3) along with their corresponding EVPN routing instances.

Figure 52: Pseudowire Terminating into Single-Homed PE Device



Benefits of Pseudowire Termination at an EVPN

- Resiliency—When a redundant logical tunnel is configured on a separate FPC, connectivity for the pseudowire automatically switches to another FPC.
- Reduced cost—Rather than using additional devices to terminate a pseudowire tunnels, you can configure pseudowire termination on the same device that also supports EVPN.

Support for Junos Node Slicing

Pseudowire termination is supported on multiple partitions in a single MX Series. Using Junos Node Slicing, you can create multiple partitions on a single MX router. These partitions are referred to as a guest network functions (GNFs). For more information on Junos Node slicing, see *Junos Node Slicing Feature Guide*.

Configuring Pseudowire Termination

To configure a pseudowire logical interface on the PE device, perform these tasks:

- Configure a logical interface.
- Configure the transport logical interface
- Configure the service logical interface.
- Configure the EVPN routing instance to support the incoming VLAN services.

To configure the logical interface:

1. Specify that you want to configure the pseudowire logical interface device.

```
user@host# edit interfaces ps0
```

2. Specify the logical tunnel interface that is the anchor point for the pseudowire logical interface device. In this case, the logical tunnel interface and anchor point is in the format **lt-*fpc/pic/port***.

NOTE: Tunnel services must be enabled in order to create the **lt** interface that is the anchor point or a member link in a redundant logical tunnel. You use the command, **set chassis fpc *slot-number* pic *pic-number* tunnel-services bandwidth *bandwidth*** to enable tunnel services.

```
[edit interfaces ps0]
user@host# set anchor-point lt-1/0/10
```

3. (Optional) Specify the MAC address for the pseudowire logical interface device.

NOTE: You should ensure that you change the MAC address prior to passing traffic or binding subscribers on the pseudowire port. Changing the MAC address when the pseudowire port is active (for example, while an upper layer protocol is negotiating) can negatively impact network performance until adjacencies learn of the new MAC address.

```
[edit interfaces ps0]
user@host# set mac 00:00:5E:00:53:55
```

4. (Optional) Specify the VLAN tagging method used for the pseudowire logical interface device. You can specify single tagging, dual (stacked) tagging, mixed (flexible) tagging, or no tagging.

```
[edit interfaces ps0]
user@host# set flexible-vlan-tagging
```

See *Enabling VLAN Tagging* for additional information about VLAN tagging.

To configure the logical interface:

1. Specify that you want to configure the pseudowire logical interface.

```
user@host# edit interfaces ps0
```

2. Specify that you want to configure unit 0 , which represents the transport logical interface.

```
[edit interfaces ps0]
user@host# edit unit 0
```

3. Specify either the encapsulation method for the transport logical interface. You can specify either **ethernet-ccc** (port-based) or **vlan-ccc** (vlan-based) encapsulation.

```
[edit interfaces ps0 unit 0]
user@host# set encapsulation ethernet-ccc
```

To configure the service logical interface:

1. Configure the unit for the service logical interface. Use a non-zero unit number.

```
user@host# edit interfaces ps0 unit1
```

2. Configure the encapsulation on the service interface.

```
[edit interfaces ps0 unit 1]
```

```
user@host# set encapsulation vlan-bridge;
```

3. (Optional) Configure the VLAN IDs.

```
[edit interfaces ps0 unit 1]
user@host# set vlan-id vlan-id;
```

4. (Optional) Configure the VLAN tag IDs to support dual-tagged VLANs.

```
[edit interfaces ps0 unit 1]
set vlan-tags inner vlan-id outer vlan-id;
```

The following is a sample configuration for a logical interface with services supporting single-tag and dual-tag VLANs and the L2 circuit associated with the interface:

```
ps0 {
  anchor-point {
    lt-0/0/0;
  }
  flexible-vlan-tagging;
  mac 00:00:5E:00:53:00;
  unit 0 {
    encapsulation ethernet-ccc;
  }
  unit 1 {
    encapsulation vlan-bridge;
    vlan-id 600;
  }
  unit 2 {
    encapsulation vlan-bridge;
    vlan-tag outer 200 inner 101;
  }
}
protocols {
  l2circuit {
    neighbor 192.168.100.1 {
      interface ps0.0 {
        virtual-circuit-id 700;
        encapsulation-type ethernet-vlan;
        ignore-mtu-mismatch;
        pseudowire-status-tlv;
      }
    }
  }
}
```

The following is a sample configuration for an EVPN routing instance that corresponds to a service logical interface:

```
routing-instances evpn-1 {
```

```
    instance-type evpn;
    vlan-id 600;
    interface ps0.1;
    route-distanguisher 3;3;
    vrf-target target:1:1;
    protocols {
      evpn;
    }
}
```

For more information on configuring logical interface termination, see *Pseudowire Subscriber Logical Interfaces Overview*.

For more information on configuring EVPN routing instances, see "Configuring EVPN Routing Instances" on page 3.

Support for Redundant Logical Tunnel

You can configure two logical tunnels on two different line cards to create a redundant logical tunnel (RLT). This provides redundancy when there is a failure in the FPC. Pseudowire over RLT supports two members in active-standby mode. Only one member link is active and carrying traffic at any given time.

To create the RLT, configure a pair of **redundant-logical-tunnel** interface at the **[edit chassis redundancy-group interface-type]** hierarchy and include the logical tunnel interface in the redundancy group by configuring **member-interface interface-name** at the **[edit interfaces interface-name redundancy-group]** hierarchy level.

The following is a sample configuration for pseudowire RLT.

```
chassis {
  pseudowire-service {
    device-count 500;
  }
  redundancy-group {
    interface-type {
      redundant-logical-tunnel {
        device-count 10;
      }
    }
  }
}
interfaces {
  rlt0 {
    redundancy-group {
      member-interface lt-0/1/0;
      member-interface lt-0/1/1;
    }
  }
  ps0 {
    anchor-point {
      rlt0;
    }
```

```
    }
}
```

For more information on redundant logical tunnels, see *Redundant Logical Tunnels Overview*

CHAPTER 18

Configuring the Distribution of Routes

Configuring an IGP on the PE and P Routers on EX9200 Switches

For Layer 2 VPNs, Layer 3 VPNs, virtual-router routing instances, VPLS, EVPNs, and Layer 2 circuits to function properly, the service provider's PE and P routers must be able to exchange routing information. For this to happen, you must configure either an IGP (such as OSPF or IS-IS) or static routes on these routers. You configure the IGP on the master instance of the routing protocol process at the **[edit protocols]** hierarchy level, not within the routing instance used for the VPN—that is, not at the **[edit routing-instances]** hierarchy level.

When you configure the PE router, do not configure any summarization of the PE router's loopback addresses at the area boundary. Each PE router's loopback address should appear as a separate route.

Related Documentation

- *Understanding IS-IS Configuration*
- *Example: Configuring IS-IS*
- *OSPF Feature Guide*

Configuring IBGP Sessions Between PE Routers in VPNs on EX9200 Switches

You must configure an IBGP session between the PE routers to allow the PE routers to exchange information about routes originating and terminating in the VPN. The PE routers rely on this information to determine which labels to use for traffic destined for remote sites.

Configure an IBGP session for the VPN as follows:

```
[edit protocols]
bgp {
   group group-name {
      type internal;
      local-address ip-address;
      family evpn {
         signaling;
      }
      family (inet-vpn | inet6-vpn) {
         unicast;
      }
      family l2vpn {
         signaling;
      }
      neighbor ip-address;
   }
}
```

The IP address in the **local-address** statement is the address of the loopback interface on the local PE router. The IBGP session for the VPN runs through the loopback address. (You must also configure the loopback interface at the **[edit interfaces]** hierarchy level.)

The IP address in the **neighbor** statement is the loopback address of the neighboring PE router.

The **family** statement allows you to configure the IBGP session for Layer 2 VPNs, VPLS, EVPNs or for Layer 3 VPNs.

- To configure an IBGP session for Layer 2 VPNs and VPLS, include the **signaling** statement at the **[edit protocols bgp group *group-name* family l2vpn]** hierarchy level:

  ```
  [edit protocols bgp group group-name family l2vpn]
  signaling;
  ```

- To configure an IBGP session for EVPNs, include the **signaling** statement at the **[edit protocols bgp group *group-name* family evpn]** hierarchy level:

  ```
  [edit protocols bgp group group-name family evpn]
  signaling;
  ```

- To configure an IPv4 IBGP session for Layer 3 VPNs, configure the **unicast** statement at the **[edit protocols bgp group *group-name* family inet-vpn]** hierarchy level:

  ```
  [edit protocols bgp group group-name family inet-vpn]
  unicast;
  ```

- To configure an IPv6 IBGP session for Layer 3 VPNs, configure the **unicast** statement at the **[edit protocols bgp group *group-name* family inet6-vpn]** hierarchy level:

  ```
  [edit protocols bgp group group-name family inet6-vpn]
  unicast;
  ```

NOTE: You can configure both **family inet** and **family inet-vpn** or both **family inet6** and **family inet6-vpn** within the same peer group. This allows you to enable support for both IPv4 and IPv4 VPN routes or both IPv6 and IPv6 VPN routes within the same peer group.

Related Documentation

- Configuring a Signaling Protocol and LSPs for VPNs on EX9200 Switches on page 571

Configuring a Signaling Protocol and LSPs for VPNs on EX9200 Switches

For VPNs to function, you must enable the LDP signaling protocol on the provider edge (PE) routers and on the provider (P) routers.

To enable the LDP signaling protocol, perform the steps in the following section:

- Using LDP for VPN Signaling on page 571

Using LDP for VPN Signaling

To use LDP for VPN signaling, perform the following steps on the PE and provider (P) routers:

1. Configure LDP on the interfaces in the core of the network by including the **ldp** statement at the **[edit protocols]** hierarchy level.

   You need to configure LDP only on the interfaces between PE routers or between PE and P routers. You can think of these as the “core-facing” interfaces. You do not need to configure LDP on the interface between the PE and customer edge (CE) routers.

   ```
   [edit]
   protocols {
     ldp {
       interface type-fpc/pic/port;
     }
   }
   ```

2. Configure the MPLS address family on the interfaces on which you enabled LDP (the interfaces you configured in Step 1) by including the **family mpls** statement at the **[edit interfaces *type-fpc/pic/port* unit *logical-unit-number*]** hierarchy level.

   ```
   [edit]
   interfaces {
     type-fpc/pic/port {
       unit logical-unit-number {
         family mpls;
       }
     }
   }
   ```

3. Configure OSPF or IS-IS on each PE and P router.

   You configure these protocols at the master instance of the routing protocol, not within the routing instance used for the VPN.

   - To configure OSPF, include the **ospf** statement at the **[edit protocols]** hierarchy level. At a minimum, you must configure a backbone area on at least one of the router's interfaces.

     ```
     [edit]
     protocols {
       ospf {
         area 0.0.0.0 {
           interface type-fpc/pic/port;
         }
       }
     }
     ```

   - To configure IS-IS, include the **isis** statement at the **[edit protocols]** hierarchy level and configure the loopback interface and International Organization for Standardization (ISO) family at the **[edit interfaces]** hierarchy level. At a minimum, you must enable IS-IS on the router, configure a network entity title (NET) on one of the router's interfaces (preferably the loopback interface, lo0), and configure the ISO family on all interfaces on which you want IS-IS to run. When you enable IS-IS, Level 1 and Level 2 are enabled by default. The following is the minimum IS-IS configuration. In the **address** statement, ***address*** is the NET.

     ```
     [edit]
     interfaces {
       lo0 {
         unit logical-unit-number {
           family iso {
             address address;
           }
         }
       }
       type-fpc/pic/port {
         unit logical-unit-number {
           family iso;
         }
       }
     }
     protocols {
       isis {
         interface all;
       }
     }
     ```

   For more information about configuring OSPF and IS-IS, see the *OSPF Feature Guide* and *IS-IS Feature Guide*.

Related Documentation

- EVPN Overview for Switches on page 552

Configuring Entropy Labels

An entropy label is a special label that enhances a network device's ability to load-balance traffic across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs). When you configure an entropy label at the ingress device, the transit device use the label to determine a path in place of performing a deep packet inspection (DPI). If the transit device does not support LDP Entropy Label Capability (ELC), the transit device ignores the entropy label and continues to propagate the packet using traditional DPI for load balancing. The entropy label is popped at the penultimate hop device.

An entropy label can be any label value between 16 to 1048575 (regular 20-bit label range). Since this range overlaps with the existing regular label range, a special label called entropy label indicator (ELI) is also inserted in the label stack before the entropy label. IANA assigned ELI label a value of 7. This distinguish entropy labels from other service labels.

To configure entropy labels, include the **entropy label** at the **[edit protocols ldp]** hierarchy on the ingress device and include the configured policy in the routing policy.

```
protocols {
  ldp {
    entropy-label {
      ingress-policy policy-name;
    }
  }
}
```

```
policy-options {
  policy-statement policy-name {
    term Add-Entropy-label {
      from {
        route-filter route exact;
        }
      then accept;
    }
  }
}
```

To enable a penultimate hop device to pop the entropy label, include the **load-balance-label-capability** statement at the **[edit forwarding-option]** hierarchy.

```
forwarding-options {
  load-balance-label-capability;
}
```

Related Documentation

- *Configuring the Entropy Label for LSPs*
- *entropy-label*

Understanding P2MPs LSP for the EVPN Inclusive Provider Tunnel

An EVPN instance comprises Customer Edge devices (CEs) that are connected to Provider Edge devices (PEs) to form the edge of the MPLS infrastructure. A CE may be a host, a router, or a switch. The PEs provide virtual Layer 2 bridged connectivity between the CEs. There may be multiple EVPN instances in the provider's network. The PEs may be connected by an MPLS Label Switched Path (LSP) infrastructure, which provides the benefits of MPLS technology, such as fast reroute, resiliency, etc.

The PEs may also be connected by an IP infrastructure, where IP/GRE (Generic Routing Encapsulation) tunneling or other IP tunneling can be used between the PEs. Here we understand the MPLS LSPs as the tunneling technology designed to be extensible to IP tunneling as the Packet Switched Network (PSN) tunneling technology.

Starting in Junos OS Release 18.2R1 onwards, Junos OS provides the ability to configure and signal a P2MP LSP for the EVPN Inclusive Provider Tunnel for BUM traffic. P2MP LSPs can provide efficient core bandwidth utilization by using the multicast replication only at the required nodes instead of ingress replication at the ingress PE.

You can configure and signal a P2MP LSP for the EVPN Inclusive Provider Tunnel in the PMSI Attributes of the Inclusive Multicast Ethernet Tag Route. The P2MP tunnel is used for forwarding Broadcast, unknown Unicast, and Multicast (BUM) packets at the ingress PE. When representing the PMSI attributes in the Inclusive Multicast Ethernet Tag Route, a transport label is not represented for P2MP Provider Tunnels, except in the case where Aggregation is used. The transport label is used to identify the EVI at the egress PE.

When the P2MP Provider Tunnels are used, the ESI labels for Split Horizon are assigned upstream instead of downstream. The label that the egress PE receives as a Split Horizon label is allocated by the ingress PE. Since each upstream PE cannot allocate the same label, the label does not identify the ESI and must be referred in the context label space of the ingress PE. The transport label uniquely identifies the ingress PE and the split horizon label is identified by tranport-label or SH-label tuple.

At the ingress PE, an upstream assigned ESI label is allocated and signaled for Split Horizon function. This label is added to the BUM traffic that originates from the given Ethernet Segment. By default, an egress may receive traffic with an ESI label that is not configured on this PE because of P2MP tunnels and upstream assigned labels.

NOTE: **For IR tunnels, ingress includes the ESI label to only those PEs with the ES. In such case, the egress PE pops the ESI label and floods the packet normally.**

For E-tree, the leaf label is assigned upstream and its function is similar to the SH label. The ingress PE allocates the leaf label and represents it in the E-tree extended community for the Ethernet A-D per ES route. When the Leaf PE acts as an ingress, it adds the leaf label for BUM traffic. An egress PE acting as a leaf discards the traffic which contains the leaf label. Egress PEs acting as a root pops the leaf label and forwards the traffic. Since leaf labels are upstream assigned, they may not be unique because multiple Leaf

PEs may have allocated the same leaf label. Therefore, the leaf label must be identified by the (tranport-label, Leaf-label) tuple.

NOTE: The EVPN P2MP functions without a lsi or vt interface. Instead, the EVPN allocates a label for use as the mLDP/RSVP transport label at the egress PE.

NSR and Unified ISSU Support on EVPN with P2MP

Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover. When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets.

Uniified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU. Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover.

When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets. Unified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU.

Junos OS mirrors essential data when NSR is enabled. For EVPN with P2MP mLDP replication, the LDP/RSVP transport label will be mirrored on the standby Routing Engine. For information on other mirrored data and NSR data flow, see "NSR and Unified ISSU Support for EVPN Overview" on page 235.

Benefits of EVPN P2MPs LSP for the EVPN Inclusive Provider Tunnel

- Provides efficient core bandwidth utilization by using multicast replication only at the required nodes.
- Manages the ingress replication at the ingress PE to avoid an over-load.
- Supports P2MP LSPs for EVPN Inclusive P-Multicast Trees for EVPNoMPLS for both mLDP and RSVP-TE P2MP-E-tree.

Related Documentation

- *Example: Configuring Point-to-Multipoint LDP LSPs as the Data Plane for Intra-AS MBGP MVPNs*
- *Example: Configuring Ingress Replication for IP Multicast Using MBGP MVPNs*
- *Flooding Unknown Traffic Using Point-to-Multipoint LSPs in VPLS*
- *Configuring RSVP-Signaled Inclusive Point-to-Multipoint LSPs for an MBGP MVPN*
- *Configuring Dynamic Point-to-Multipoint Flooding LSPs*

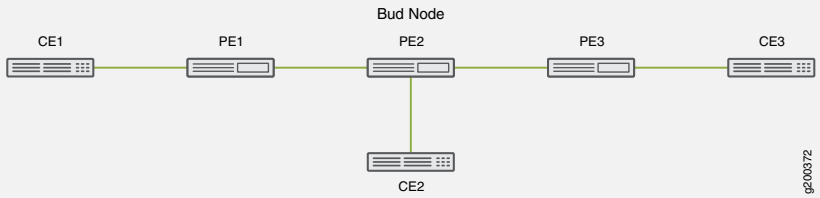
- *Configuring RSVP Automatic Mesh*
- *container-label-switched-path*

Configuring Bud Node Support

Along a LSP, PE devices function as an ingress device, transit device or egress device. A bud node is a PE device that functions as both an egress and transit device. In a P2MP LSP, you can have devices that will function in the role of a both as a transit device and an egress device.

Figure 53 on page 576 illustrates a simple EVPN network with a bud node at PE2. When CE1 sends a multicast packet out, PE2 operates as a transit device and forwards the packet to PE3 . It also functions as a egress device and pops the MPLS label and replicates multicast packets destined for CE2.

Figure 53: Bud Node in an EVPN Network



To enable a PE device to function as a bud-node, include **p2mp-bud-support** statement at the **[edit routing-instances routing-instance-name protocols evpn]** hierarchy. When **p2mp-bud-support** is enabled or disabled, you may observed dropped packets on the device. This occurs because changing bud node support affects the forwarding state in the routing instance, which results in the forwarding table being rebuilt.

NOTE: We recommend that all PE devices in the LSP that may potentially function as both a egress and transit device be enabled as a bud node.

CHAPTER 19

Configuring VLAN Bundle Services, VLAN-Aware Bundle Services, and Virtual Switch Support

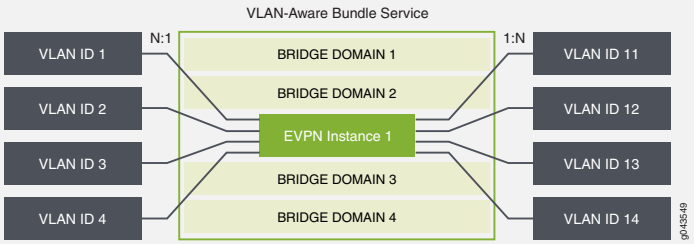
Overview of VLAN Bundle Service and VLAN-Aware Bundle Service for EVPN

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. When deploying the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

The DCSP might require Ethernet VLAN services to be extended over a WAN with a single EVPN instance (EVI). Junos OS supports bothVLAN bundle service and VLAN-aware bundle service, which maintain data and control plane separation. Figure 54 on page 578 illustrates the relationship between VLANs and EVIs in VLAN bundle service and VLAN-aware bundle service. Both support the mapping of one EVI to many VLAN IDs (VIDs), but VLAN bundle service supports only one bridge domain and the bridge table is shared among the different VLANs, while VLAN-aware bundle service supports multiple bridge tables, with each bridge table corresponding to a different VLAN.

Figure 54: VLAN Bundle Service and VLAN-Aware Bundle Service



Related Documentation

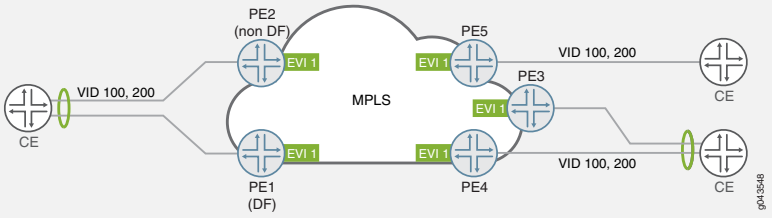
- VLAN Bundle Service for EVPN on page 578
- Virtual Switch Support for EVPN Overview on page 289

VLAN Bundle Service for EVPN

Starting with Junos OS Release 17.1, VLAN bundle service allows multiple broadcast domains to map to a single bridge domain. Multiple VLANs are mapped to a single EVPN instance (EVI) and share the same bridge table in the MAC-VRF table, thus reducing the number of routes and labels stored in the table. This lowers the control plane overhead on the router. Having a single bridge domain requires all CE devices in the VLAN network to have unique MAC addresses. VLAN ID translation is not permitted as the MPLS encapsulated frames must retain the originating VLAN ID. As such the Ethernet Tag ID in all EVPN routes is set to zero. A VLAN range cannot be specified. The entire VLAN from 1 to 4095 must be bundled on the interface.

Figure 55 on page 578 illustrates a topology where VID 100 and VID 200 are bundled and assigned to EVI 1. The service provider creates a single broadcast domain for the customer and assigns a preconfigured number of CE-VIDs on the ingress PE routers (PE1 through PE5) to EVI 1. All CE devices use the same CE-VIDs for the EVI.

Figure 55: VLAN Bundle Network Topology



Using the VLAN bundle service reduces the number of routes and labels, which in turn, reduces the control plane overhead. The trade-off for the ease of provisioning in a customer network is that the service provider has no control over the customer broadcast domain since there is a single inclusive multicast tree and no CE-VID translation. .

NOTE: Junos OS also supports port-based VLAN bundle service where all of the VLANs on a port are part of the same service and are mapped to the same bundle.

NOTE: Integrated routing and bridging (IRB) is not supported for VLAN bundle service.

Release History Table

Release	Description
17.1	Starting with Junos OS Release 17.1, VLAN bundle service allows multiple broadcast domains to map to a single bridge domain.

Related Documentation

- Configuring EVPN VLAN Bundle Services on page 579

Configuring EVPN VLAN Bundle Services

To configure EVPN VLAN bundle services , complete the following configuration on all PE routers within the EVPN service provider's network:

1. Configure the EVPN routing instance name using the **routing-instances** statement at the **[edit]** hierarchy level:

   routing-instances *routing-instance-name* {...}

2. Configure the **evpn** option for the **instance-type** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

   instance-type evpn;

3. Configure a route distinguisher on a PE router by including the **route-distinguisher** statement:

   route-distinguisher (*as-number:number* | *ip-address:number*);

   Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN routing instances need a route distinguisher to help BGP to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs. If you configure different VPN routing instances with the same route distinguisher, the commit fails.

   For a list of the hierarchy levels at which you can include the **route-distinguisher** statement, see route-distinguisher.

The route distinguisher is a 6-byte value that you can specify in either of the following formats:

- ***as-number*:*number***, where ***as-number*** is an autonomous system (AS) number (a 2-byte value) and ***number*** is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the ISP's own or the customer's own AS number.

  NOTE: The automatic derivation of the BGP route target (auto-RT) for advertised prefixes is supported on 2-byte AS numbers only.

- ***ip-address*:*number***, where ***ip-address*** is an IP address (a 4-byte value) and ***number*** is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the **router-id** statement, which is a nonprivate address in your assigned prefix range.

4. Configure either import and export policies for the EVPN routing table, or configure the default policies using the **vrf-target** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level.

   See *Configuring Policies for the VRF Table on PE Routers in VPNs*.

5. Configure each EVPN interface for the EVPN routing instance:

   NOTE: Adding a trunk port with dual tags to an EVPN and MPLS routing instance to an EVPN and VXLAN routing instance causes the CLI commit check configuration to fail and generate an error. To avoid configuration check errors, use sub-interface style interface configuration with dual tags for the routing instances.

   a. Configure each interface using the interface statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

   b. Configure interface encapsulation for the CE-facing interfaces at the **[edit interfaces *interface-name* encapsulation]** hierarchy level. Supported encapsulations are ethernet-bridge, vlan-bridge, and extended-vlan-bridge. .

      NOTE: If you are configuring the port-based VLAN bundle service, you will need to also configure the interface encapsulation ethernet-bridge unit to 0.

   c. (Optional) Include the *ignore-encapsulation-mismatch* statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level to allow the EVPN to establish a connection to the CE device even

if the CE device interface encapsulation and the EVPN interface encapsulations do not match.

d. Specify a static MAC address for a logical interface in a bridge domain using the *static-mac* statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level.

6. Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN routing instance by including the interface-mac-limit statement.

   You can configure the same limit for all interfaces configured for a routing instance by including the **interface-mac-limit** statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level. You can also configure a limit for a specific interface by including this statement at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level.

   By default, packets with new source MAC addresses are forwarded after the MAC address limit is reached. You can alter this behavior by including the **packet-action drop** statement at either the **[edit routing-instances *routing-instance-name* protocols evpn interface-mac-limit]** or the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level. If you configure this statement, packets from new source MAC addresses are dropped after the configured MAC address limit is reached.

7. Specify the MPLS label allocation setting for the EVPN by including the *label-allocation* statement with the **per-instance** option at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

   If you configure this statement, one MPLS label is allocated for the specified EVPN routing instance.

8. Enable MAC accounting for the EVPN by including the *mac-statistics* statement at the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level.

9. Disable MAC learning by including the *no-mac-learning* statement at either the **[edit routing-instances *routing-instance-name* protocols evpn]** hierarchy level to apply this behavior to all of the devices configured for an EVPN routing instance or at the **[edit routing-instances *routing-instance-name* protocols evpn interface *interface-name*]** hierarchy level to apply this behavior to only one of the CE devices.

The following output shows a sample EVPN VLAN bundle services configuration.

```
user@router1# show routing-instances evpn_1
```

```
instance-type evpn;
          interface xe-2/3/3.300;
          route-distinguisher 7.7.7.7:3;
          vrf-target target:65221:3;
          protocols evpn;
          label-allocation per-instance;
```

```
        }
    }
```

Related Documentation

- VLAN Bundle Service for EVPN on page 578

Virtual Switch Support for EVPN Overview

Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers. Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches. Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches. . This feature allows Ethernet VLANs over a WAN to share a single EVPN instance while maintaining data-plane separation between the different VLANs.

Junos OS has a highly flexible and scalable virtual switch interface. With a virtual switch, a single router or switch can be divided into multiple logical switches. Layer 2 domains (also called *bridge-domains* or *vlans*) can be defined independently in each virtual switch. To configure VLAN-aware bundle service, an EVPN must run in a virtual-switch routing instance.

On the EX Series and MX Series, a single EVPN instance can stretch up to 4094 bridge domains or VLANs defined in a virtual switch to remote sites. A virtual switch can have more than 4094 bridge domains or VLANs with a combination of none, single, and dual VLANs. However, because EVPN signaling deals only with single VLAN tags, a maximum of 4094 bridge domains or VLANs can be stretched. The EVPN virtual switch also provides support for trunk and access interfaces.

On the QFX10000 switches, Layer 2 VLANs use the default-switch routing instance. An EVPN instance (EVI) is not supported.

NOTE:

- The none VLAN option is supported with bridge domains or VLANs under the virtual switch instance type for EVPNs.
- Access interfaces configured with single or dual VLAN tags are supported in EVPN. By default, only Ethernet frames with single or no VLAN tags are transported across the EVPN core. As a result, dual-tagged Ethernet frames received on the access interfaces must be normalized to Ethernet frames with single or no VLAN tags for proper transmission over the EVPN core.

  You can enable transporting of dual-tagged frames across the EVPN core network by including both the vlan-id none and no-normalization configuration statements together.

There are two types of VLAN-aware bundle service:

- VLAN-aware bundle without translation

The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance with a guarantee end-to-end customer VLAN transparency. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN.

NOTE: **The QFX10000 switches only support VLAN-aware bundle service interface without translation. Support is limited to 4K VLANs because only the default-switching instance is supported.**

- VLAN-aware bundle with translation

  The service interface provides bundling of customer VLANs into a single Layer 2 VPN service instance. The data-plane separation between the customer VLANs is maintained by creating a dedicated bridge-domain for each VLAN. The service interface supports customer VLAN translation to handle the scenario where different VLAN Identifiers (VIDs) are used on different interfaces to designate the same customer VLAN.

EVPN with virtual switch provides support for VLAN-aware bundle with translation only.

Release History Table

Release	Description
17.3	Starting with Junos OS Release 17.3, VLAN-aware bundle service is introduced on QFX Series switches.
14.2	Starting with Junos OS Release 14.2, VLAN-aware bundle service is introduced on EX9200 switches.
14.1	Starting with Junos OS Release 14.1, VLAN-aware bundle service is introduced on MX Series routers.

Related Documentation

- Example: Configuring EVPN with Support for Virtual Switch on page 587

Configuring EVPN with Support for Virtual Switch

You can configure an Ethernet VPN (EVPN) with virtual switch support to enable multiple tenants with independent VLAN and subnet space within an EVPN instance. Virtual switch provides the ability to extend Ethernet VLANs over a WAN using a single EVPN instance while maintaining data-plane separation between the various VLANs associated with that instance. A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites.

When configuring virtual switch for EVPN, be aware of the following considerations:

- Due to default ARP policing, some of the ARP packets not destined for the device can be missed. This can lead to delayed ARP learning and synchronization.
- Clearing ARP for an EVPN can lead to inconsistency between the ARP table and the EVPN ARP table. To avoid this situation, clear both ARP and EVPN ARP tables.
- The **vlan-tag** can be configured for local switching. However, vlan-tagged VLANs should not be extended over the EVPN cloud.

Before you begin:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Enable chained composite next hop for EVPN.
4. Configure OSPF or any other IGP protocol.
5. Configure a BGP internal group.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure RSVP or LDP.
8. Configure MPLS.
9. Create a label-switched path between the provider edge (PE) devices.

To configure the PE device:

1. Configure the virtual switch routing instance.

   ```
   [edit routing-instances]
   user@PE1# set evpn-instance instance-type virtual-switch
   ```

2. Configure the interface names for the virtual switch routing instance.

   ```
   [edit routing-instances]
   user@PE1# set evpn-instance interface interface-name
   ```

3. Configure the route distinguisher for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

4. Configure the VPN routing and forwarding (VRF) target community for the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target
```

5. List the VLAN identifiers that are to be EVPN extended.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn extended-vlan-list [vlan-id-range]
```

6. Configure the bridge domain for the first virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name domain-type
  bridge
```

7. Assign the VLAN ID for the first bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name vlan-id 10
```

8. Configure the IRB interface as the routing interface for the first bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name
  routing-interface irb.0
```

9. Configure the interface name for the first bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains first-bridge-domain-name bridge-options
  interface CE-facing-interface
```

10. Configure the bridge domain for the second virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name
  domain-type bridge
```

11. Assign the VLAN ID for the second bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name vlan-id
  VLAN-ID
```

12. Configure the IRB interface as the routing interface for the second bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name
  routing-interface irb.1
```

13. Configure the interface name for the second bridge domain.

```
[edit routing-instances]
user@PE1# set evpn-instance bridge-domains second-bridge-domain-name
  bridge-options interface CE-facing-interface
```

14. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
```

15. Configure the IRB interface as the routing interface for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance interface irb.0
user@PE1# set vrf-instance interface irb.1
```

16. Configure the route distinguisher for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
```

17. Configure the VRF target community for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-target vrf-target
```

18. Configure the VRF label for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-table-label
```

19. Verify and commit the configuration.

    For example:

```
[edit routing-instances]
user@PE1# set evpna instance-type virtual-switch
user@PE1# set evpna interface ge-0/1/4.0
user@PE1# set evpna interface ge-0/1/4.1
user@PE1# set evpna route-distinguisher 10.255.169.37:1
user@PE1# set evpna vrf-target target:100:1
user@PE1# set evpna protocols evpn extended-vlan-list [ 10 20 ]
user@PE1# set evpna bridge-domains bda domain-type bridge
user@PE1# set evpna bridge-domains bda vlan-id 10
user@PE1# set evpna bridge-domains bda routing-interface irb.0
user@PE1# set evpna bridge-domains bda bridge-options interface ge-0/1/4.0
user@PE1# set evpna bridge-domains bdb domain-type bridge
user@PE1# set evpna bridge-domains bdb vlan-id 20
user@PE1# set evpna bridge-domains bdb routing-interface irb.1
user@PE1# set evpna bridge-domains bdb bridge-options interface ge-0/1/4.1
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface irb.0
user@PE1# set vrf interface irb.1
user@PE1# set vrf route-distinguisher 192.0.2.1:2
user@PE1# set vrf vrf-target target:100:2
user@PE1# set vrf vrf-table-label
```

```
[edit]
user@PE1# commit
commit complete
```

Related Documentation

- Example: Configuring EVPN with Support for Virtual Switch on page 587

Example: Configuring EVPN with Support for Virtual Switch

- Example: Configuring EVPN with Support for Virtual Switch on page 587

Example: Configuring EVPN with Support for Virtual Switch

This example shows how to configure a virtual switch in an Ethernet VPN (EVPN) deployment.

Requirements

This example uses the following hardware and software components:

- Two MX Series 5G Universal Routing Platforms containing MPC FPCs.
- Two customer edge (CE) routers.
- Junos OS Release 14.1 or later.

Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure RSVP or LDP.
5. Configure MPLS.

Overview

Starting with Junos OS Release 14.1, the Ethernet VPN (EVPN) solution on MX Series routers with MPC interfaces is extended to provide virtual switch support that enables multiple tenants with independent VLAN and subnet space within an EVPN instance. Virtual switch provides the ability to extend Ethernet VLANs over a WAN using a single EVPN instance while maintaining data-plane separation between the various VLANs associated with that instance. A single EVPN instance can stretch up to 4094 bridge domains defined in a virtual switch to remote sites.

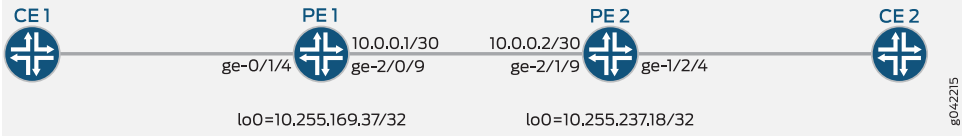
When configuring virtual switch for EVPN, be aware of the following considerations:

- Due to default ARP policing, some of the ARP packets not destined for the device can be missed. This can lead to delayed ARP learning and synchronization.
- Clearing ARP for an EVPN can lead to inconsistency between the ARP table and the EVPN ARP table. To avoid this situation, clear both ARP and EVPN ARP tables.
- The **vlan-tag** can be configured for local switching. However, vlan-tagged VLANs should not be extended over the EVPN cloud.

Topology

Figure 56 on page 588 illustrates a simple EVPN topology with virtual switch support. Routers PE1 and PE2 are the provider edge (PE) routers that connect to one customer edge (CE) router each – CE1 and CE2.

Figure 56: EVPN with Virtual Switch Support



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

PE1

```
set interfaces ge-2/0/9 unit 0 family inet address 10.0.0.1/30
set interfaces ge-2/0/9 unit 0 family mpls
set interfaces ge-0/1/4 flexible-vlan-tagging
set interfaces ge-0/1/4 encapsulation flexible-ethernet-services
set interfaces ge-0/1/4 unit 0 family bridge interface-mode trunk
set interfaces ge-0/1/4 unit 0 vlan-id-list 10
set interfaces ge-0/1/4 unit 1 family bridge interface-mode trunk
set interfaces ge-0/1/4 unit 1 vlan-id-list 20
set interfaces irb unit 0 family inet address 192.168.1.1/16
set interfaces irb unit 1 family inet address 192.168.2.1/16
set interfaces lo0 unit 0 family inet address 10.255.169.37/32
set routing-options router-id 10.255.169.37
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE1-to-PE2 from 10.255.169.37
set protocols mpls label-switched-path PE1-to-PE2 to 10.255.237.18
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.169.37
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.237.18
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type virtual-switch
set routing-instances evpna interface ge-0/1/4.0
set routing-instances evpna interface ge-0/1/4.1
set routing-instances evpna route-distinguisher 10.255.169.37:1
set routing-instances evpna vrf-target target:100:1
set routing-instances evpna protocols evpn extended-vlan-list [ 10 20 ]
set routing-instances evpna bridge-domains bda domain-type bridge
set routing-instances evpna bridge-domains bda vlan-id 10
set routing-instances evpna bridge-domains bda routing-interface irb.0
set routing-instances evpna bridge-domains bda bridge-options interface ge-0/1/4.0
set routing-instances evpna bridge-domains bdb domain-type bridge
set routing-instances evpna bridge-domains bdb vlan-id 20
set routing-instances evpna bridge-domains bdb routing-interface irb.1
set routing-instances evpna bridge-domains bdb bridge-options interface ge-0/1/4.1
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf route-distinguisher 198.51.100.1:2
set routing-instances vrf vrf-target target:100:2
set routing-instances vrf vrf-table-label
```

PE2

```
set interfaces ge-2/1/9 unit 0 family inet address 10.0.0.2/30
set interfaces ge-2/1/9 unit 0 family mpls
set interfaces ge-1/2/4 flexible-vlan-tagging
set interfaces ge-1/2/4 encapsulation flexible-ethernet-services
set interfaces ge-1/2/4 unit 0 family bridge interface-mode trunk
set interfaces ge-1/2/4 unit 0 vlan-id-list 10
```

```
set interfaces ge-1/2/4 unit 1 family bridge interface-mode trunk
set interfaces ge-1/2/4 unit 1 vlan-id-list 20
set interfaces irb unit 0 family inet address 192.168.2.2/16
set interfaces irb unit 1 family inet address 192.168.2.3/16
set interfaces lo0 unit 0 family inet address 10.255.237.18/32
set routing-options router-id 10.255.237.18
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE2-to-PE1 from 10.255.237.18
set protocols mpls label-switched-path PE2-to-PE1 to 10.255.169.37
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.237.18
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.169.37
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type virtual-switch
set routing-instances evpna interface ge-1/2/4.0
set routing-instances evpna interface ge-1/2/4.1
set routing-instances evpna route-distinguisher 10.255.237.18:1
set routing-instances evpna vrf-target target:100:1
set routing-instances evpna protocols evpn extended-vlan-list [ 10 20 ]
set routing-instances evpna bridge-domains bda domain-type bridge
set routing-instances evpna bridge-domains bda vlan-id 10
set routing-instances evpna bridge-domains bda routing-interface irb.0
set routing-instances evpna bridge-domains bda bridge-options interface ge-1/2/4.0
set routing-instances evpna bridge-domains bdb domain-type bridge
set routing-instances evpna bridge-domains bdb vlan-id 20
set routing-instances evpna bridge-domains bdb routing-interface irb.1
set routing-instances evpna bridge-domains bdb bridge-options interface ge-1/2/4.1
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf interface irb.1
set routing-instances vrf route-distinguisher 198.51.100.2:2
set routing-instances vrf vrf-target target:100:2
set routing-instances vrf vrf-table-label
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2, after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the PE1 interfaces.

```
[edit interfaces]
user@PE1# set ge-2/0/9 unit 0 family inet address 10.0.0.1/30
user@PE1# set ge-2/0/9 unit 0 family mpls
user@PE1# set ge-0/1/4 flexible-vlan-tagging
user@PE1# set ge-0/1/4 encapsulation flexible-ethernet-services
user@PE1# set ge-0/1/4 unit 0 family bridge interface-mode trunk
user@PE1# set ge-0/1/4 unit 0 vlan-id-list 10
user@PE1# set ge-0/1/4 unit 1 family bridge interface-mode trunk
user@PE1# set ge-0/1/4 unit 1 vlan-id-list 20
user@PE1# set irb unit 0 family inet address 192.168.1.1/16
user@PE1# set irb unit 1 family inet address 192.168.2.1/16
user@PE1# set lo0 unit 0 family inet address 10.255.169.37/32
```

2. Set the router ID and autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set router-id 10.255.169.37
user@PE1# set autonomous-system 100
```

3. Configure the chained composite next hop for EVPN.

```
[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn
```

4. Enable RSVP on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
```

5. Create label-switched paths for PE1 to reach PE2.

```
[edit protocols]
user@PE1# set mpls label-switched-path PE1-to-PE2 from 10.255.169.37
user@PE1# set mpls label-switched-path PE1-to-PE2 to 10.255.237.18
```

6. Enable MPLS on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable
```

7. Configure the BGP group for Router PE1.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
```

8. Assign local and neighbor addresses to the ibgp BGP group for Router PE1 to peer with Router PE2.

```
[edit protocols]
user@PE1# set bgp group ibgp local-address 10.255.169.37
user@PE1# set bgp group ibgp neighbor 10.255.237.18
```

9. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@PE1# set bgp group ibgp family evpn signaling
```

10. Configure OSPF on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

11. Configure the virtual switch routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type virtual-switch
```

12. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna interface ge-0/1/4.0
user@PE1# set evpna interface ge-0/1/4.1
```

13. Configure the route distinguisher for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna route-distinguisher 10.255.169.37:1
```

14. Configure the VPN routing and forwarding (VRF) target community for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vrf-target target:100:1
```

15. List the VLAN identifiers that are to be EVPN extended.

```
[edit routing-instances]
user@PE1# set evpna protocols evpn extended-vlan-list [ 10 20 ]
```

16. Configure the bridge domains for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda domain-type bridge
```

17. Assign the VLAN ID for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda vlan-id 10
```

18. Configure the IRB interface as the routing interface for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda routing-interface irb.0
```

19. Configure the interface name for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bda bridge-options interface ge-0/1/4.0
```

20. Configure the bridge domains for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb domain-type bridge
```

21. Assign the VLAN ID for the bdb bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb vlan-id 20
```

22. Configure the IRB interface as the routing interface for the bda bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb routing-interface irb.1
```

23. Configure the interface name for bdb bridge domain.

```
[edit routing-instances]
user@PE1# set evpna bridge-domains bdb bridge-options interface ge-0/1/4.1
```

24. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
```

25. Configure the IRB interface as the routing interface for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf interface irb.0
user@PE1# set vrf interface irb.1
```

26. Configure the route distinguisher for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 198.51.100.1:2
```

27. Configure the VRF target community for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-target target:100:2
```

28. Configure VRF label for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-2/0/9 {
   unit 0 {
      family inet {
         address 10.0.0.1/30;
      }
      family mpls;
   }
}
ge-0/1/4 {
   flexible-vlan-tagging;
   encapsulation flexible-ethernet-services;
   unit 0 {
      family bridge {
         interface-mode trunk;
         vlan-id-list 10;
      }
   }
   unit 1 {
      family bridge {
         interface-mode trunk;
```

```
        vlan-id-list 20;
      }
    }
  }
  irb {
    unit 0 {
      family inet {
        address 192.168.1.1/16;
      }
    }
    unit 1 {
      family inet {
        address 192.168.2.1/16;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.169.37/32;
      }
    }
  }
```

```
user@PE1# show routing-options
router-id 10.255.169.37;
autonomous-system 100;
forwarding-table {
  chained-composite-next-hop {
    ingress {
      evpn;
    }
  }
}
```

```
user@PE1# show protocols
rsvp {
  interface all;
  interface fxp0.0 {
    disable;
  }
}
mpls {
  label-switched-path PE1-to-PE2 {
    from 10.255.169.37;
    to 10.255.237.18;
  }
  interface all;
  interface fxp0.0 {
    disable;
  }
}
bgp {
  group ibgp {
```

```
      type internal;
      local-address 10.255.169.37;
      family evpn {
        signaling;
      }
      neighbor 10.255.237.18;
    }
  }
  ospf {
    area 0.0.0.0 {
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
  }
```

```
user@PE1# show routing-instances
evpna {
    instance-type virtual-switch;
    interface ge-0/1/4.0;
    interface ge-0/1/4.1;
    route-distinguisher 10.255.169.37:1;
    vrf-target target:100:1;
    protocols {
        evpn {
            extended-vlan-list [ 10 20 ];
        }
    }
    bridge-domains {
        bda {
            domain-type bridge;
            vlan-id 10;
            routing-interface irb.0;
            bridge-options {
                interface ge-0/1/4.0;
            }
        }
        bdb {
            domain-type bridge;
            vlan-id 20;
            routing-interface irb.1;
            bridge-options {
                interface ge-0/1/4.1;
            }
        }
    }
}
vrf {
    instance-type vrf;
    interface irb.0;
    interface irb.1;
    route-distinguisher 10.255.169.37:2;
    vrf-target target:100:2;
```

```
    vrf-table-label;
}
```

Verification

Confirm that the configuration is working properly.

Verifying the Bridge Domain Configuration

Purpose Verify the bridge domain configuration for the evpna routing instance.

Action From operational mode, run the **show bridge domain extensive** command.

```
user@PE1> show bridge domain extensive

Routing instance: evpna
Bridge domain: bda                                State: Active
Bridge VLAN ID: 10                                EVPN extended: Yes
Interfaces:
    ge-0/1/4.0
    pip-10.000010000000
    pip-10.feff0f000000
Total MAC count: 2

Bridge domain: bdb                                State: Active
Bridge VLAN ID: 20                                EVPN extended: Yes
Interfaces:
    ge-0/1/4.1
    pip-11.010010000000
    pip-11.ffff0f000000
Total MAC count: 2
```

Meaning The configured bridge domains **bda** and **bdb** and their associated VLAN IDs and interfaces are displayed. The bridge domains are also extended with EVPN.

Verifying MAC Table Routes

Purpose Verify the MACs learned in the data plane and control plane.

Action From operational mode, run the **show bridge mac-table** command.

```
user@PE1> show bridge mac-table

MAC flags (S -static MAC, D -dynamic MAC, L -locally learned, C -Control MAC
           SE -Statistics enabled, NM -Non configured MAC, R -Remote PE MAC)

Routing instance : evpna
 Bridging domain : bda, VLAN : 10
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:00:00:aa:01:01   S        ge-0/1/4.0
   00:00:00:bb:01:01   DC                        1048574 1048574
   00:00:00:cc:01:01   DC                        1048576 1048576

Bridging domain : bdb, VLAN : 20
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:00:00:aa:02:01   S        ge-0/1/4.1
   00:00:00:bb:02:01   DC                        1048575 1048575
   00:00:00:cc:02:01   DC                        1048577 1048577
```

Meaning The configured static MACs for the bridge domains are displayed.

Verifying the Bridge EVPN Peer Gateway MAC

Purpose Verify the bridge EVPN peer gateway MAC for the evpna routing instance.

Action From operational mode, run the **show bridge evpn peer-gateway-macs** command.

```
user@PE1> show bridge evpn peer-gateway-macs

Routing instance : evpna
 Bridging domain : bda, VLAN : 10
  Installed GW MAC addresses:
  00:23:9c:96:af:f0
  a8:d0:e5:5b:02:08

Bridging domain : bdb, VLAN : 20
  Installed GW MAC addresses:
  00:23:9c:96:af:f0
  a8:d0:e5:5b:02:08
```

Meaning The gateway MACs of the EVPN peers for the evpna routing instance are displayed.

See Also
- Virtual Switch Support for EVPN Overview on page 289

CHAPTER 20

Configuring Integrated Bridging and Routing

EVPN with IRB Solution Overview

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. To deploy the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN (EVPN) is targeted to handle all of the above mentioned challenges, wherein:

- The basic EVPN functionality enables optimal intra-subnet traffic forwarding
- Implementing the integrated routing and bridging (IRB) solution in an EVPN deployment enables optimal inter-subnet traffic forwarding
- Configuring EVPN with virtual switch support enables multiple tenants with independent VLAN and subnet space

The following sections describe the IRB solution for EVPNs:

- Need for an EVPN IRB Solution on page 600
- Implementing the EVPN IRB Solution on page 601
- Benefits of Implementing the EVPN IRB Solution on page 603

Need for an EVPN IRB Solution

EVPN is a technology used to provide Layer 2 extension and interconnection across an IP/MPLS core network to different physical sites belonging to a single Layer 2 domain. In a data center environment with EVPN, there is a need for both Layer 2 (intra-subnet traffic) and Layer 3 (inter-subnet traffic) forwarding and potentially interoperation with tenant Layer 3 VPNs.

With only a Layer 2 solution, there is no optimum forwarding of inter-subnet traffic, even when the traffic is local, for instance, when both the subnets are on the same server.

With only a Layer 3 solution, the following issues for intra-subnet traffic can arise:

- MAC address aliasing issue where duplicate MAC addresses are not detected.
- TTL issue for applications that use TTL 1 to confine traffic within a subnet.
- IPv6 link-local addressing and duplicate address detection that relies on Layer 2 connectivity.

- Layer 3 forwarding does not support the forwarding semantics of a subnet broadcast.
- Support of non-IP applications that require Layer 2 forwarding.

Because of the above mentioned shortcomings of a pure Layer 2 and Layer 3 solution, there is a need for a solution incorporating optimal forwarding of both Layer 2 and Layer 3 traffic in the data center environment when faced with operational considerations such as Layer 3 VPN interoperability and virtual machine (VM) mobility.

An EVPN-based integrated routing and bridging (IRB) solution provides optimum unicast and multicast forwarding for both intra-subnets and inter-subnets within and across data centers.

The EVPN IRB feature is useful for service providers operating in an IP/MPLS network that provides both Layer 2 VPN or VPLS services and Layer 3 VPN services who want to extend their service to provide cloud computation and storage services to their existing customers.

Implementing the EVPN IRB Solution

An EVPN IRB solution provides the following:

- Optimal forwarding for intra-subnet (Layer 2) traffic.
- Optimal forwarding for inter-subnet (Layer 3) traffic.
- Support for ingress replication for multicast traffic.
- Support for network-based as well as host-based overlay models.
- Support for consistent policy-based forwarding for both Layer 2 and Layer 3 traffic.

Junos OS supports several models of EVPN configuration to satisfy the individual needs of EVPN and data center cloud services customers. To provide flexibility and scalability, multiple bridge domains can be defined within a particular EVPN instance. Likewise, one or more EVPN instances can be associated with a single Layer 3 VPN virtual routing and forwarding (VRF). In general, each data center tenant is assigned a unique Layer 3 VPN VRF, while a tenant could comprise one or more EVPN instances and one or more bridge domains per EVPN instance. To support this model, each configured bridge domain (including the default bridge domain for an EVPN instance) requires an IRB interface to perform the Layer 2 and Layer 3 functions. Each bridge domain or IRB interface maps to a unique IP subnet in the VRF.

NOTE: **You can associate an IRB interface with the master instance inet.0 table instead of a VRF in an EVPN IRB solution.**

There are two major functions that are supported for IRB in EVPN.

- Host MAC-IP synchronization

This includes:

- Advertising the IP address along with the MAC advertisement route in EVPN. This is done by using the IP field in the EVPN MAC advertisement route.
- The receiving PE router installs MAC into the EVPN instance (EVI) table and installs IP into the associated VRF.

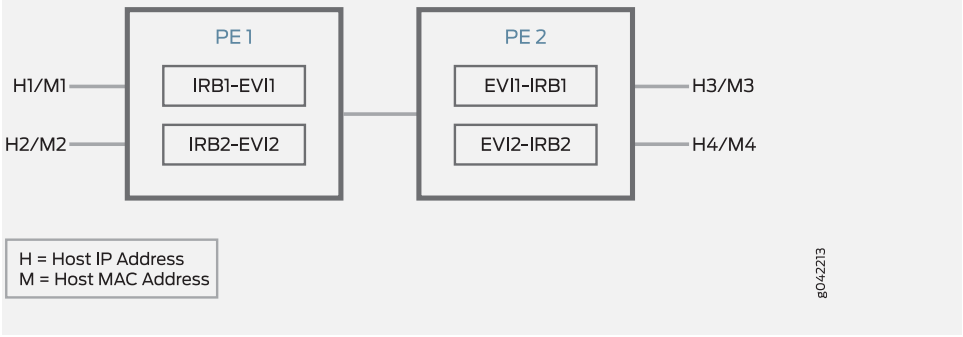
- Gateway MAC-IP synchronization

  This includes:

  - Advertising all local IRB MAC and IP addresses in an EVPN. This is achieved by including the default gateway extended community in the EVPN MAC advertisement route.
  - The receiving PE creates a forwarding state to route packets destined for the gateway MAC, and a proxy ARP is done for the gateway IP with the MAC advertised in the route.

Figure 57 on page 602 illustrates the inter-subnet traffic forwarding between two provider edge (PE) devices – PE1 and PE2. The IRB1 and IRB2 interfaces on each PE device belong to a different subnet, but they share a common VRF.

Figure 57: Inter-Subnet Traffic Forwarding



The inter-subnet traffic forwarding is performed as follows:

1. PE2 advertises H3-M3 and H4-M4 binding to PE1. Similarly PE1 advertises H1-M1 and H2-M2 binding to PE2.

2. PE1 and PE2 install the MAC address in the corresponding EVI MAC table, whereas the IP routes are installed in the shared VRF.

3. The advertising PE device is set as the next hop for the IP routes.

4. If H1 sends packets to H4, the packets are sent to IRB1 on PE1.

5. IP lookup for H4 happens in the shared VRF on PE1. Because the next hop for the H4 IP is PE2 (the advertising PE), an IP unicast packet is sent to PE2.

6. PE1 rewrites the MAC header based on the information in the VRF route, and PE2 performs a MAC lookup to forward the packet to H4.

Benefits of Implementing the EVPN IRB Solution

The main goal of the EVPN IRB solution is to provide optimal Layer 2 and Layer 3 forwarding. The solution is required to efficiently handle inter-subnet forwarding as well as virtual machine (VM) mobility. VM mobility refers to the ability of a VM to migrate from one server to another within the same or a different data center while retaining its existing MAC and IP address. Providing optimal forwarding for inter-subnet traffic and effective VM mobility involves solving two problems – the default gateway problem and the triangular routing problem.

Starting in Junos OS Release 17.1R1, IPv6 addresses are supported on IRB interfaces with EVPN using the Neighbor Discovery Protocol (NDP). The following capabilities are introduced for IPv6 support with EVPN:

- IPv6 addresses on IRB interfaces in master routing instances
- Learning IPv6 neighborhood from solicited NA message
- NS and NA packets on the IRB interfaces are disabled from network core
- Virtual gateway addresses are used as Layer 3 addresses
- Host MAC-IP synchronization for IPv6

You can configure the IPv6 addresses in the IRB interface at the **[edit interfaces irb]** hierarchy level.

- Gateway MAC and IP Synchronization on page 603
- Layer 3 VPN Interworking on page 604

Gateway MAC and IP Synchronization

In an EVPN IRB deployment, the IP default gateway for a VM is the IP address configured on the IRB interface of the provider edge (PE) router corresponding to the bridge domain or VLAN of which the VM is a member. The default gateway problem arises because a VM does not flush its ARP table when relocating from one server to another and continues sending packets with the destination MAC address set to that of the original gateway. If the old and new servers are not part of the same Layer 2 domain (the new Layer 2 domain could be within the current data center or a new data center), the gateway previously identified is no longer the optimal or local gateway. The new gateway needs to identify packets containing the MAC addresses of other gateways on remote PE routers and forward the traffic as if the packets were destined to the local gateway itself. At the minimum, this functionality requires each PE router to advertise its gateway or IRB MAC and IP addresses to all other PE routers in the network. The gateway address exchange can be accomplished using the standard MAC route advertisement message (including the IP address parameter) and tagging that route with the default gateway extended

community so that the remote PE routers can distinguish the gateway MAC advertisement routes from normal MAC advertisement routes.

Layer 3 VPN Interworking

The inter-data center aspect of the EVPN IRB solution involves routing between VMs that are present in different data centers or routing between a host site completely outside of the data center environment and a VM within a data center. This solution relies on the ability of EVPN MAC route advertisements to carry both MAC address and IP address information. The local MAC learning functionality of the PE router is extended to also capture IP address information associated with MAC addresses learned locally. That IP-MAC address mapping information is then distributed to each PE router through normal EVPN procedures. When a PE router receives such MAC and IP information, it installs the MAC route in the EVPN instance as well as a host route for the associated IP address in the Layer 3 VPN VRF corresponding to that EVPN instance. When a VM moves from one data center to another, normal EVPN procedures result in the MAC and IP address being advertised from the new PE router which the VM resides behind. The host route installed in the VRF associated with an EVPN solicits Layer 3 traffic destined to that VM to the new PE router and avoids triangular routing between the source, the former PE router the VM resided behind, and the new PE router.

BGP scalability is a potential concern with the inter-data center triangular routing avoidance solution because of the potential for injection of many host routes into Layer 3 VPN. With the method previously described, in the worst case there is an IP host route for each MAC address learned through the local EVPN MAC learning procedures or through a MAC advertisement message received from a remote PE router. BGP route target filtering can be used to limit distribution of such routes.

The following functional elements are required to implement the inter-data center triangular routing avoidance using Layer 3 inter-subnet forwarding procedures:

1. The source host sends an IP packet using its own source MAC and IP address with the destination MAC of the IRB interface of the local PE router and the IP address of the destination host.

2. When the IRB interface receives the frame with its MAC as the destination, it performs a Layer 3 lookup in the VRF associated with the EVPN instance to determine where to route the packet.

3. In the VRF, the PE router finds the Layer 3 route derived from a MAC plus an IP EVPN route received from the remote PE router earlier. The destination MAC address is then changed to the destination MAC address corresponding to the destination IP.

4. The packet is then forwarded to the remote PE router serving the destination host using MPLS, using the label corresponding to the EVPN instance of which the destination host is a member.

5. The egress PE router receiving the packet performs a Layer 2 lookup for the destination host's MAC and sends the packet to the destination host on the attached subnet via the egress PE router's IRB interface.

6. Because the ingress PE router is performing Layer 3 routing, the IP TTL is decremented.

Related Documentation

- EVPN Multihoming Overview on page 29
- Understanding VXLANs on page 260
- Example: Configuring EVPN with IRB Solution on page 620

An EVPN with IRB Solution on EX9200 Switches Overview

A data center service provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span the data center over multiple sites. To deploy data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM) motion.
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN (EVPN) is targeted to handle all the preceding challenges, wherein:

- The basic EVPN functionality enables optimal intra-subnet traffic forwarding
- Implementing the integrated routing and bridging (IRB) solution in an EVPN deployment enables optimal inter-subnet traffic forwarding
- Configuring EVPN with virtual switch support enables multiple tenants with independent VLAN and subnet space

The following sections describe the integrated routing and bridging (IRB) solution for EVPNs:

- Need for an EVPN IRB Solution on page 606
- Implementing the EVPN IRB Solution on page 606
- Benefits of Implementing the EVPN IRB Solution on page 608
- IPv6 Support for IRB Interfaces with EVPN Using Neighborhood Discovery Protocol (NDP) on page 610

Need for an EVPN IRB Solution

EVPN is a technology used to provide Layer 2 extension and interconnection across an IP/MPLS core network to different physical sites belonging to a single Layer 2 domain. In a data center environment with EVPN, there is a need for both Layer 2 (intra-subnet traffic) and Layer 3 (inter-subnet traffic) forwarding and potentially interoperation with tenant Layer 3 VPNs.

With only a Layer 2 solution, there is no optimum forwarding of inter-subnet traffic, even when the traffic is local, for instance, when both the subnets are on the same server.

With only a Layer 3 solution, the following issues for intra-subnet traffic can arise:

- MAC address aliasing issue where duplicate MAC addresses are not detected.
- TTL issue for applications that use TTL 1 to confine traffic within a subnet.
- IPv6 link-local addressing and duplicate address detection that relies on Layer 2 connectivity.
- Layer 3 forwarding does not support the forwarding semantics of a subnet broadcast.
- Support of non-IP applications that require Layer 2 forwarding.

Because of the above mentioned shortcomings of a pure Layer 2 and Layer 3 solution, there is a need for a solution incorporating optimal forwarding of both Layer 2 and Layer 3 traffic in the data center environment when faced with operational considerations such as Layer 3 VPN interoperability and virtual machine (VM) mobility.

An EVPN-based integrated routing and bridging (IRB) solution provides optimum unicast and multicast forwarding for both intra-subnets and inter-subnets within and across data centers.

The EVPN IRB feature is useful for service providers operating in an IP/MPLS network that provides both Layer 2 VPN or VPLS services and Layer 3 VPN services who want to extend their service to provide cloud computation and storage services to their existing customers.

Implementing the EVPN IRB Solution

An EVPN IRB solution provides the following:

- Optimal forwarding for intra-subnet (Layer 2) traffic.
- Optimal forwarding for inter-subnet (Layer 3) traffic.
- Support for ingress replication for multicast traffic.
- Support for network-based as well as host-based overlay models.
- Support for consistent policy-based forwarding for both Layer 2 and Layer 3 traffic.

Junos OS supports several models of EVPN configuration to satisfy the individual needs of EVPN and data center cloud services customers. To provide flexibility and scalability, multiple VLANs can be defined within a particular EVPN instance. Likewise, one or more

EVPN instances can be associated with a single Layer 3 VPN virtual routing and forwarding (VRF). In general, each data center tenant is assigned a unique Layer 3 VPN VRF, while a tenant could comprise one or more EVPN instances and one or more VLANs per EVPN instance. To support this model, each configured VLAN (including the default VLAN for an EVPN instance) requires an IRB interface to perform the Layer 2 and Layer 3 functions. Each VLAN or IRB interface maps to a unique IP subnet in the VRF.

There are two major functions that are supported for IRB in EVPN.

- Host MAC-IP synchronization

  This includes:

  - Advertising the IP address along with the MAC advertisement route in EVPN. This is done by using the IP field in the EVPN MAC advertisement route.
  - The receiving PE router installs MAC into the EVPN instance (EVI) table and installs IP into the associated VRF.

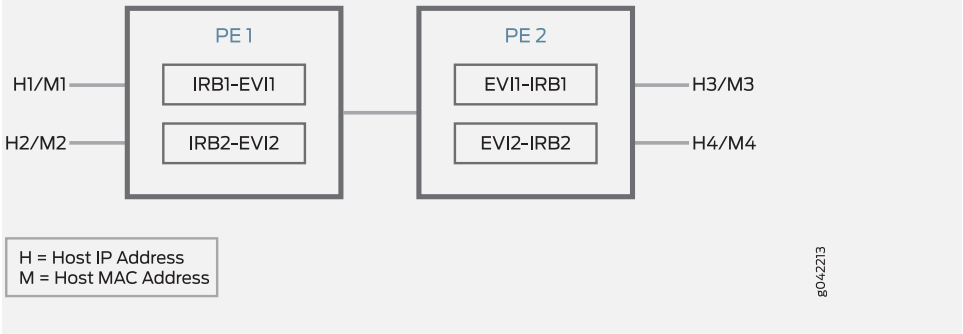
- Gateway MAC-IP synchronization

  This includes:

  - Advertising all local IRB MAC and IP addresses in an EVPN. This is achieved by including the default gateway extended community in the EVPN MAC advertisement route.
  - The receiving PE creates a forwarding state to route packets destined for the gateway MAC, and a proxy ARP is done for the gateway IP with the MAC advertised in the route.

Figure 57 on page 602 illustrates the inter-subnet traffic forwarding between two provider edge (PE) devices—PE1 and PE2. The IRB1 and IRB2 interfaces on each PE device belong to a different subnet, but they share a common VRF.

Figure 58: Inter-Subnet Traffic Forwarding



The inter-subnet traffic forwarding is performed as follows:

1. PE2 advertises the H3-M3 and H4-M4 binding to PE1. Similarly, PE1 advertises the H1-M1 and H2-M2 binding to PE2.
2. PE1 and PE2 install the MAC address in the corresponding EVI MAC table, whereas the IP routes are installed in the shared VRF.
3. The advertising PE device is set as the next hop for the IP routes.
4. If H1 sends packets to H4, the packets are sent to IRB1 on PE1.
5. IP lookup for H4 happens in the shared VRF on PE1. Because the next hop for the H4 IP is PE2 (the advertising PE), an IP unicast packet is sent to PE2.
6. PE1 rewrites the MAC header based on the information in the VRF route, and PE2 performs a MAC lookup to forward the packet to H4.

Benefits of Implementing the EVPN IRB Solution

The main goal of the EVPN IRB solution is to provide optimal Layer 2 and Layer 3 forwarding. The solution is required to efficiently handle inter-subnet forwarding as well as virtual machine (VM) mobility. VM mobility refers to the ability of a VM to migrate from one server to another within the same or a different data center while retaining its existing MAC and IP address. Providing optimal forwarding for inter-subnet traffic and effective VM mobility involves solving two problems – the default gateway problem and the triangular routing problem.

- Gateway MAC and IP Synchronization on page 608
- Layer 3 VPN Interworking on page 609

Gateway MAC and IP Synchronization

In an EVPN IRB deployment, the IP default gateway for a VM is the IP address configured on the IRB interface of the provider edge (PE) router corresponding to the VLAN of which the VM is a member. The default gateway problem arises because a VM does not flush its ARP table when relocating from one server to another and continues sending packets with the destination MAC address set to that of the original gateway. If the old and new servers are not part of the same Layer 2 domain (the new Layer 2 domain could be within the current data center or a new data center), the gateway previously identified is no longer the optimal or local gateway. The new gateway needs to identify packets containing the MAC addresses of other gateways on remote PE routers and forward the traffic as if the packets were destined to the local gateway itself. At minimum, this functionality requires each PE router to advertise its gateway or IRB MAC and IP addresses to all other PE routers in the network. The gateway address exchange can be accomplished using the standard MAC route advertisement message (including the IP address parameter) and tagging that route with the default gateway extended community so that the remote

PE routers can distinguish the gateway MAC advertisement routes from normal MAC advertisement routes.

Layer 3 VPN Interworking

The inter-data center aspect of the EVPN IRB solution involves routing between VMs that are present in different data centers or routing between a host site completely outside of the data center environment and a VM within a data center. This solution relies on the ability of EVPN MAC route advertisements to carry both MAC address and IP address information. The local MAC learning functionality of the PE router is extended to also capture IP address information associated with MAC addresses learned locally. That IP-MAC address mapping information is then distributed to each PE router through normal EVPN procedures. When a PE router receives such MAC and IP information, it installs the MAC route in the EVPN instance as well as a host route for the associated IP address in the Layer 3 VPN VRF corresponding to that EVPN instance. When a VM moves from one data center to another, normal EVPN procedures result in the MAC and IP address being advertised from the new PE router which the VM resides behind. The host route installed in the VRF associated with an EVPN solicits Layer 3 traffic destined to that VM to the new PE router and avoids triangular routing between the source, the former PE router the VM resided behind, and the new PE router.

BGP scalability is a potential concern with the inter-data center triangular routing avoidance solution because of the potential for injection of many host routes into Layer 3 VPN. With the method previously described, in the worst case there is an IP host route for each MAC address learned through the local EVPN MAC learning procedures or through a MAC advertisement message received from a remote PE router. BGP route target filtering can be used to limit distribution of such routes.

The following functional elements are required to implement the inter-data center triangular routing avoidance using Layer 3 inter-subnet forwarding procedures:

1. The source host sends an IP packet using its own source MAC and IP address with the destination MAC of the IRB interface of the local PE router and the IP address of the destination host.

2. When the IRB interface receives the frame with its MAC as the destination, it performs a Layer 3 lookup in the VRF associated with the EVPN instance to determine where to route the packet.

3. In the VRF, the PE router finds the Layer 3 route derived from a MAC plus an IP EVPN route received from the remote PE router earlier. The destination MAC address is then changed to the destination MAC address corresponding to the destination IP.

4. The packet is then forwarded to the remote PE router serving the destination host using MPLS, using the label corresponding to the EVPN instance of which the destination host is a member.

5. The egress PE router receiving the packet performs a Layer 2 lookup for the destination host's MAC and sends the packet to the destination host on the attached subnet via the egress PE router's IRB interface.

6. Because the ingress PE router is performing Layer 3 routing, the IP TTL is decremented.

IPv6 Support for IRB Interfaces with EVPN Using Neighborhood Discovery Protocol (NDP)

Starting in Junos OS Release 17.3R1, IPv6 addresses are supported on IRB interfaces with EVPN using NDP. The following capabilities are introduced for IPv6 support with EVPN:

- IPv6 addresses on IRB interfaces in master routing instances
- Learning IPv6 neighbors from solicited neighbor advertisement (NA) messages
- Neighbor solicitation (NS) and neighbor advertisement (NA) packets on the IRB interfaces are disabled from network core
- Virtual gateway addresses are used as Layer 3 addresses
- Host MAC-IP synchronization for IPv6

You can configure the IPv6 addresses in the IRB interface at the **[edit interfaces irb]** hierarchy level.

Release History Table

Release	Description
17.3R1	Starting in Junos OS Release 17.3R1, IPv6 addresses are supported on IRB interfaces with EVPN

Related Documentation

- Example: Configuring an EVPN with IRB Solution on EX9200 Switches on page 636
- EVPN Overview for Switches on page 552

Configuring EVPN with IRB Solution

You can configure an Ethernet VPN (EVPN) with IRB solution to enable Layer 2 switching and Layer 3 routing operations within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant virtual routing and forwarding (VRF) routing instances.

Before you begin:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Enable chained composite next hop for EVPN.
4. Configure OSPF or any other IGP protocol.

5. Configure a BGP internal group.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure RSVP or LDP.
8. Configure MPLS.
9. Create a label-switched path between the provider edge (PE) devices.

To configure the PE device:

1. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type evpn
```

2. Set the VLAN identifier for the bridging domain in the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vlan-id VLAN-ID
```

3. Configure the interface name for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface CE-facing-interface
```

4. Configure the IRB interface as the routing interface for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance routing-interface irb.0
```

5. Configure the route distinguisher for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

6. Configure the VPN routing and forwarding (VRF) target community for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target-value
```

7. Assign the interface name that connects the PE device site to the VPN.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn interface CE-facing-interface
```

8. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
```

9. Configure the IRB interface as the routing interface for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance interface irb.0
```

10. Configure the route distinguisher for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
```

11. Configure the VRF label for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-table-label
```

12. Verify and commit the configuration.

    For example:

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
user@PE1# set evpna vlan-id 10
user@PE1# set evpna interface ge-1/1/8.0
user@PE1# set evpna routing-interface irb.0
user@PE1# set evpna route-distinguisher 192.0.2.1:100
user@PE1# set evpna vrf-target target:100:100
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface irb.0
user@PE1# set vrf route-distinguisher 192.0.2.1:300
user@PE1# set vrf vrf-target target:100:300
user@PE1# set vrf vrf-table-label
```

```
[edit]
user@PE1# commit
commit complete
```

Related Documentation

- Example: Configuring EVPN with IRB Solution on page 615

Configuring an EVPN with IRB Solution on EX9200 Switches

You can configure an Ethernet VPN (EVPN) with IRB solution to enable Layer 2 switching and Layer 3 routing operations within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant virtual routing and forwarding (VRF) routing instances.

Before you begin:

1. Configure the switch interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Enable the chained composite next hop for EVPN.
4. Configure OSPF or any other IGP protocol.
5. Configure a BGP internal group.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure LDP.
8. Configure MPLS.

To configure the PE device:

1. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance instance-type evpn
```

2. Set the VLAN identifier for the bridging domain in the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vlan-id VLAN-ID
```

3. Configure the interface name for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance interface CE-facing-interface
```

4. Configure the IRB interface as the routing interface for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance l3-interface irb.0
```

5. Configure the route distinguisher for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance route-distinguisher route-distinguisher-value
```

6. Configure the VPN routing and forwarding (VRF) target community for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn-instance vrf-target vrf-target-value
```

7. Assign the interface name that connects the PE device site to the VPN.

```
[edit routing-instances]
user@PE1# set evpn-instance protocols evpn interface CE-facing-interface
```

8. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance instance-type vrf
```

9. Configure the IRB interface as the routing interface for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance interface irb.0
```

10. Configure the route distinguisher for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance route-distinguisher route-distinguisher-value
```

11. Configure the VRF label for the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf-instance vrf-table-label
```

12. Verify and commit the configuration.

    For example:

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
user@PE1# set evpna vlan-id 10
user@PE1# set evpna interface ge-1/1/8.0
user@PE1# set evpna l3-interface irb.0
user@PE1# set evpna route-distinguisher 100.255.0.1:100
user@PE1# set evpna vrf-target target:100:100
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
user@PE1# set vrf instance-type vrf
```

```
user@PE1# set vrf interface irb.0
user@PE1# set vrf route-distinguisher 100.255.0.1:300
user@PE1# set vrf vrf-target target:100:300
user@PE1# set vrf vrf-table-label
```

```
[edit]
user@PE1# commit
commit complete
```

Related Documentation

- Example: Configuring an EVPN with IRB Solution on EX9200 Switches on page 636

Example: Configuring EVPN with IRB Solution

EVPN with IRB Solution Overview

A Data Center Service Provider (DCSP) hosts the data center for its multiple customers on a common physical network. To each customer (also called a tenant), the service looks like a full-fledged data center that can expand to 4094 VLANs and all private subnets. For disaster recovery, high availability, and optimization of resource utilization, it is common for the DCSP to span across the data center to more than one site. To deploy the data center services, a DCSP faces the following main challenges:

- Extending Layer 2 domains across more than one data center site. This requires optimal intra-subnet traffic forwarding.
- Supporting optimal inter-subnet traffic forwarding and optimal routing in the event of virtual machine (VM).
- Supporting multiple tenants with independent VLAN and subnet space.

Ethernet VPN (EVPN) is targeted to handle all of the above mentioned challenges, wherein:

- The basic EVPN functionality enables optimal intra-subnet traffic forwarding
- Implementing the integrated routing and bridging (IRB) solution in an EVPN deployment enables optimal inter-subnet traffic forwarding
- Configuring EVPN with virtual switch support enables multiple tenants with independent VLAN and subnet space

The following sections describe the IRB solution for EVPNs:

Need for an EVPN IRB Solution

EVPN is a technology used to provide Layer 2 extension and interconnection across an IP/MPLS core network to different physical sites belonging to a single Layer 2 domain. In a data center environment with EVPN, there is a need for both Layer 2 (intra-subnet traffic) and Layer 3 (inter-subnet traffic) forwarding and potentially interoperation with tenant Layer 3 VPNs.

With only a Layer 2 solution, there is no optimum forwarding of inter-subnet traffic, even when the traffic is local, for instance, when both the subnets are on the same server.

With only a Layer 3 solution, the following issues for intra-subnet traffic can arise:

- MAC address aliasing issue where duplicate MAC addresses are not detected.
- TTL issue for applications that use TTL 1 to confine traffic within a subnet.
- IPv6 link-local addressing and duplicate address detection that relies on Layer 2 connectivity.
- Layer 3 forwarding does not support the forwarding semantics of a subnet broadcast.
- Support of non-IP applications that require Layer 2 forwarding.

Because of the above mentioned shortcomings of a pure Layer 2 and Layer 3 solution, there is a need for a solution incorporating optimal forwarding of both Layer 2 and Layer 3 traffic in the data center environment when faced with operational considerations such as Layer 3 VPN interoperability and virtual machine (VM) mobility.

An EVPN-based integrated routing and bridging (IRB) solution provides optimum unicast and multicast forwarding for both intra-subnets and inter-subnets within and across data centers.

The EVPN IRB feature is useful for service providers operating in an IP/MPLS network that provides both Layer 2 VPN or VPLS services and Layer 3 VPN services who want to extend their service to provide cloud computation and storage services to their existing customers.

Implementing the EVPN IRB Solution

An EVPN IRB solution provides the following:

- Optimal forwarding for intra-subnet (Layer 2) traffic.
- Optimal forwarding for inter-subnet (Layer 3) traffic.
- Support for ingress replication for multicast traffic.
- Support for network-based as well as host-based overlay models.
- Support for consistent policy-based forwarding for both Layer 2 and Layer 3 traffic.

Junos OS supports several models of EVPN configuration to satisfy the individual needs of EVPN and data center cloud services customers. To provide flexibility and scalability, multiple bridge domains can be defined within a particular EVPN instance. Likewise, one or more EVPN instances can be associated with a single Layer 3 VPN virtual routing and

forwarding (VRF). In general, each data center tenant is assigned a unique Layer 3 VPN VRF, while a tenant could comprise one or more EVPN instances and one or more bridge domains per EVPN instance. To support this model, each configured bridge domain (including the default bridge domain for an EVPN instance) requires an IRB interface to perform the Layer 2 and Layer 3 functions. Each bridge domain or IRB interface maps to a unique IP subnet in the VRF.

NOTE: **You can associate an IRB interface with the master instance inet.0 table instead of a VRF in an EVPN IRB solution.**

There are two major functions that are supported for IRB in EVPN.

- Host MAC-IP synchronization

  This includes:

  - Advertising the IP address along with the MAC advertisement route in EVPN. This is done by using the IP field in the EVPN MAC advertisement route.
  - The receiving PE router installs MAC into the EVPN instance (EVI) table and installs IP into the associated VRF.

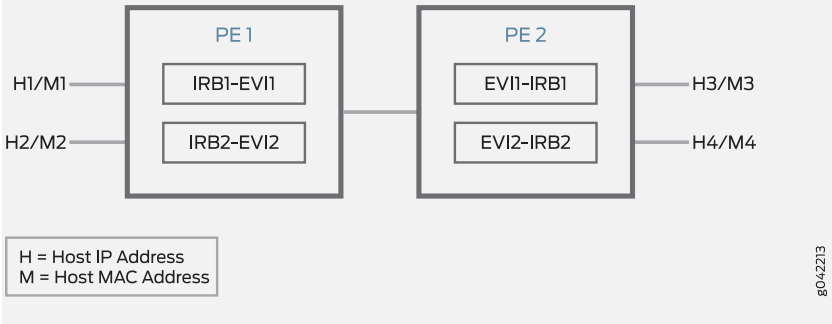
- Gateway MAC-IP synchronization

  This includes:

  - Advertising all local IRB MAC and IP addresses in an EVPN. This is achieved by including the default gateway extended community in the EVPN MAC advertisement route.
  - The receiving PE creates a forwarding state to route packets destined for the gateway MAC, and a proxy ARP is done for the gateway IP with the MAC advertised in the route.

Figure 57 on page 602 illustrates the inter-subnet traffic forwarding between two provider edge (PE) devices – PE1 and PE2. The IRB1 and IRB2 interfaces on each PE device belong to a different subnet, but they share a common VRF.

Figure 59: Inter-Subnet Traffic Forwarding



The inter-subnet traffic forwarding is performed as follows:

1. PE2 advertises H3-M3 and H4-M4 binding to PE1. Similarly PE1 advertises H1-M1 and H2-M2 binding to PE2.
2. PE1 and PE2 install the MAC address in the corresponding EVI MAC table, whereas the IP routes are installed in the shared VRF.
3. The advertising PE device is set as the next hop for the IP routes.
4. If H1 sends packets to H4, the packets are sent to IRB1 on PE1.
5. IP lookup for H4 happens in the shared VRF on PE1. Because the next hop for the H4 IP is PE2 (the advertising PE), an IP unicast packet is sent to PE2.
6. PE1 rewrites the MAC header based on the information in the VRF route, and PE2 performs a MAC lookup to forward the packet to H4.

Benefits of Implementing the EVPN IRB Solution

The main goal of the EVPN IRB solution is to provide optimal Layer 2 and Layer 3 forwarding. The solution is required to efficiently handle inter-subnet forwarding as well as virtual machine (VM) mobility. VM mobility refers to the ability of a VM to migrate from one server to another within the same or a different data center while retaining its existing MAC and IP address. Providing optimal forwarding for inter-subnet traffic and effective VM mobility involves solving two problems – the default gateway problem and the triangular routing problem.

Starting in Junos OS Release 17.1R1, IPv6 addresses are supported on IRB interfaces with EVPN using the Neighbor Discovery Protocol (NDP). The following capabilities are introduced for IPv6 support with EVPN:

- IPv6 addresses on IRB interfaces in master routing instances
- Learning IPv6 neighborhood from solicited NA message
- NS and NA packets on the IRB interfaces are disabled from network core
- Virtual gateway addresses are used as Layer 3 addresses
- Host MAC-IP synchronization for IPv6

You can configure the IPv6 addresses in the IRB interface at the **[edit interfaces irb]** hierarchy level.

- Gateway MAC and IP Synchronization on page 619
- Layer 3 VPN Interworking on page 619

Gateway MAC and IP Synchronization

In an EVPN IRB deployment, the IP default gateway for a VM is the IP address configured on the IRB interface of the provider edge (PE) router corresponding to the bridge domain or VLAN of which the VM is a member. The default gateway problem arises because a VM does not flush its ARP table when relocating from one server to another and continues sending packets with the destination MAC address set to that of the original gateway. If the old and new servers are not part of the same Layer 2 domain (the new Layer 2 domain could be within the current data center or a new data center), the gateway previously identified is no longer the optimal or local gateway. The new gateway needs to identify packets containing the MAC addresses of other gateways on remote PE routers and forward the traffic as if the packets were destined to the local gateway itself. At the minimum, this functionality requires each PE router to advertise its gateway or IRB MAC and IP addresses to all other PE routers in the network. The gateway address exchange can be accomplished using the standard MAC route advertisement message (including the IP address parameter) and tagging that route with the default gateway extended community so that the remote PE routers can distinguish the gateway MAC advertisement routes from normal MAC advertisement routes.

Layer 3 VPN Interworking

The inter-data center aspect of the EVPN IRB solution involves routing between VMs that are present in different data centers or routing between a host site completely outside of the data center environment and a VM within a data center. This solution relies on the ability of EVPN MAC route advertisements to carry both MAC address and IP address information. The local MAC learning functionality of the PE router is extended to also capture IP address information associated with MAC addresses learned locally. That IP-MAC address mapping information is then distributed to each PE router through normal EVPN procedures. When a PE router receives such MAC and IP information, it installs the MAC route in the EVPN instance as well as a host route for the associated IP address in the Layer 3 VPN VRF corresponding to that EVPN instance. When a VM moves from one data center to another, normal EVPN procedures result in the MAC and IP address being advertised from the new PE router which the VM resides behind. The host route installed in the VRF associated with an EVPN solicits Layer 3 traffic destined to that VM to the new PE router and avoids triangular routing between the source, the former PE router the VM resided behind, and the new PE router.

BGP scalability is a potential concern with the inter-data center triangular routing avoidance solution because of the potential for injection of many host routes into Layer 3 VPN. With the method previously described, in the worst case there is an IP host route for each MAC address learned through the local EVPN MAC learning procedures or through a MAC advertisement message received from a remote PE router. BGP route target filtering can be used to limit distribution of such routes.

The following functional elements are required to implement the inter-data center triangular routing avoidance using Layer 3 inter-subnet forwarding procedures:

1. The source host sends an IP packet using its own source MAC and IP address with the destination MAC of the IRB interface of the local PE router and the IP address of the destination host.

2. When the IRB interface receives the frame with its MAC as the destination, it performs a Layer 3 lookup in the VRF associated with the EVPN instance to determine where to route the packet.

3. In the VRF, the PE router finds the Layer 3 route derived from a MAC plus an IP EVPN route received from the remote PE router earlier. The destination MAC address is then changed to the destination MAC address corresponding to the destination IP.

4. The packet is then forwarded to the remote PE router serving the destination host using MPLS, using the label corresponding to the EVPN instance of which the destination host is a member.

5. The egress PE router receiving the packet performs a Layer 2 lookup for the destination host's MAC and sends the packet to the destination host on the attached subnet via the egress PE router's IRB interface.

6. Because the ingress PE router is performing Layer 3 routing, the IP TTL is decremented.

See Also

- EVPN Multihoming Overview on page 29
- Understanding VXLANs on page 260
- Example: Configuring EVPN with IRB Solution on page 620

Example: Configuring EVPN with IRB Solution

This example shows how to configure an integrated routing and bridging (IRB) solution in an Ethernet VPN (EVPN) deployment.

Requirements

This example uses the following hardware and software components:

- Two MX Series 5G Universal Routing Platforms containing MPC FPCs configured as PE routers.

- Two customer edge (CE) routers, each connected to the PE routers.
- Junos OS Release 14.1 or later running on all the PE routers.

Before you begin:

1. Configure the router interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure RSVP or LDP.
5. Configure MPLS.

Overview

In an EVPN solution, multiple bridge domains can be defined within a particular EVPN instance, and one or more EVPN instances can be associated with a single Layer 3 VPN VRF. In general, each data center tenant is assigned a unique Layer 3 VPN virtual route forwarding (VRF), although the tenant can be comprised of one or more EVPN instances or bridge domains per EVPN instance.

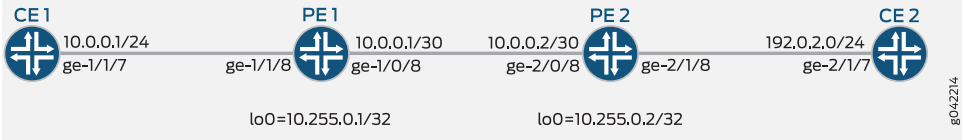
To support this flexibility and scalability factor, the EVPN solution provides support for the IRB interfaces on MX Series routers containing MPC FPCs to facilitate optimal Layer 2 and Layer 3 forwarding along with virtual machine mobility. The IRB interfaces are configured on each configured bridge domain including the default bridge domain for an EVPN instance.

IRB is the ability to do Layer 2 switching and Layer 3 routing within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant VRFs.

Topology

Figure 60 on page 621 illustrates a simple EVPN topology with IRB solution. Routers PE1 and PE2 are the provider edge routers that connect to two customer edge (CE) routers each – CE1 and CE2.

Figure 60: EVPN with IRB Solution



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network

configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

CE1

```
set interfaces ge-1/1/7 vlan-tagging
set interfaces ge-1/1/7 unit 0 vlan-id 10
set interfaces ge-1/1/7 unit 0 family inet address 10.0.0.1/24
set routing-options static route 192.0.2.0/24 next-hop 10.0.0.251
```

PE1

```
set interfaces ge-1/0/8 unit 0 family inet address 10.0.0.1/30
set interfaces ge-1/0/8 unit 0 family mpls
set interfaces ge-1/1/8 flexible-vlan-tagging
set interfaces ge-1/1/8 encapsulation flexible-ethernet-services
set interfaces ge-1/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/8 unit 0 vlan-id 10
set interfaces irb unit 0 family inet address 10.0.0.251/24
set interfaces lo0 unit 0 family inet address 10.255.0.1/32
set routing-options router-id 10.255.0.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE1-to-PE2 from 10.255.0.1
set protocols mpls label-switched-path PE1-to-PE2 to 10.255.0.2
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.0.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.0.2
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 10
set routing-instances evpna interface ge-1/1/8.0
set routing-instances evpna routing-interface irb.0
set routing-instances evpna route-distinguisher 10.255.0.1:100
set routing-instances evpna vrf-target target:100:100
set routing-instances evpna protocols evpn interface ge-1/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 10.255.0.1:300
set routing-instances vrf vrf-target target:100:300
set routing-instances vrf vrf-table-label
```

PE2

```
set interfaces ge-2/0/8 unit 0 family inet address 10.0.0.2/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces ge-2/1/8 flexible-vlan-tagging
set interfaces ge-2/1/8 encapsulation flexible-ethernet-services
set interfaces ge-2/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-2/1/8 unit 0 vlan-id 20
```

```
set interfaces irb unit 0 family inet address 192.0.2.1/24
set interfaces lo0 unit 0 family inet address 10.255.0.2/32
set routing-options router-id 10.255.0.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE2-to-PE1 from 10.255.0.2
set protocols mpls label-switched-path PE2-to-PE1 to 10.255.0.1
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.0.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 10.255.0.1
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf traffic-engineering
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 20
set routing-instances evpna interface ge-2/1/8.0
set routing-instances evpna routing-interface irb.0
set routing-instances evpna route-distinguisher 10.255.0.2:100
set routing-instances evpna vrf-target target:200:100
set routing-instances evpna protocols evpn interface ge-2/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 10.255.0.2:300
set routing-instances vrf vrf-target target:200:300
set routing-instances vrf vrf-table-label
```

CE2

```
set interfaces ge-2/1/7 unit 0 vlan-id 20
set interfaces ge-2/1/7 unit 0 family inet address 192.0.2.0/24
set routing-options static route 10.0.0.0/24 next-hop 192.0.2.5
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2, after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Router PE1 interfaces.

```
[edit interfaces]
user@PE1# set ge-1/0/8 unit 0 family inet address 10.0.0.1/30
user@PE1# set ge-1/0/8 unit 0 family mpls
```

```
user@PE1# set ge-1/1/8 flexible-vlan-tagging
user@PE1# set ge-1/1/8 encapsulation flexible-ethernet-services
user@PE1# set ge-1/1/8 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/1/8 unit 0 vlan-id 10
user@PE1# set irb unit 0 family inet address 10.0.0.251/24
user@PE1# set lo0 unit 0 family inet address 10.255.0.1/32
```

2. Set the router ID and autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set router-id 10.255.0.1
user@PE1# set autonomous-system 100
```

3. Configure the chained composite next hop for EVPN.

```
[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn
```

4. Enable RSVP on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
```

5. Create label-switched path for Router PE1 to reach Router PE2.

```
[edit protocols]
user@PE1# set mpls label-switched-path PE1-to-PE2 from 10.255.0.1
user@PE1# set mpls label-switched-path PE1-to-PE2 to 10.255.0.2
```

6. Enable MPLS on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls mpls interface fxp0.0 disable
```

7. Configure the BGP group for Router PE1.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
```

8. Assign local and neighbor addresses to the ibgp BGP group for Router PE1 to peer with Router PE2.

```
[edit protocols]
user@PE1# set bgp group ibgp local-address 10.255.0.1
user@PE1# set bgp group ibgp neighbor 10.255.0.2
```

9. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@PE1# set bgp group ibgp family evpn signaling
```

10. Configure OSPF on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

11. Enable traffic-engineering for OSPF. For RSVP-signaled LSPs with OSPF as the IGP, traffic-engineering must be enabled for the LSPs to come up.

```
[edit protocols]
user@PE1# set ospf traffic-engineering
```

12. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
```

13. Set the VLAN identifier for the bridging domain in the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vlan-id 10
```

14. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna interface ge-1/1/8.0
```

15. Configure the IRB interface as the routing interface for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna routing-interface irb.0
```

16. Configure the route distinguisher for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna route-distinguisher 10.255.0.1:100
```

17. Configure the VPN routing and forwarding (VRF) target community for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vrf-target target:100:100
```

18. Assign the interface name that connects the PE1 site to the VPN.

```
[edit routing-instances]
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
```

19. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
```

20. Configure the IRB interface as the routing interface for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf interface irb.0
```

21. Configure the route distinguisher for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 10.255.0.1:300
```

22. Configure the VRF label for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-1/0/8 {
   unit 0 {
      family inet {
```

```
        address 10.0.0.1/30;
      }
      family mpls;
    }
  }
  ge-1/1/8 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id 10;
    }
  }
  irb {
    unit 0 {
      family inet {
        address 10.0.0.251/24;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.0.1/32 {
        }
      }
    }
  }
```

```
user@PE1# show routing-options
router-id 10.255.0.1;
autonomous-system 100;
forwarding-table {
  chained-composite-next-hop {
    ingress {
      evpn;
    }
  }
}
```

```
user@PE1# show protocols
rsvp {
  interface all;
  interface fxp0.0 {
    disable;
  }
}
mpls {
  label-switched-path PE1-to-PE2 {
    from 10.255.0.1;
    to 10.255.0.2;
  }
  interface all;
  interface fxp0.0 {
```

```
      disable;
    }
  }
  bgp {
    group ibgp {
      type internal;
      local-address 10.255.0.1;
      family evpn {
        signaling;
      }
      neighbor 10.255.0.2;
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
  }
```

```
user@PE1# show routing-instances
evpna {
  instance-type evpn;
  vlan-id 10;
  interface ge-1/1/8.0;
  routing-interface irb.0;
  route-distinguisher 10.255.0.1:100;
  vrf-target target:100:100;
  protocols {
    evpn {
      interface ge-1/1/8.0;
    }
  }
}
vrf {
  instance-type vrf;
  interface irb.0;
  route-distinguisher 10.255.0.1:300;
  vrf-target target:100:300;
  vrf-table-label;
}
```

Verification

Confirm that the configuration is working properly.

- Verifying Local IRB MACs on page 629
- Verifying Remote IRB MACs on page 630
- Verifying Local IRB IPs on page 631
- Verifying Remote IRB IPs on page 632

Verifying Local IRB MACs

Purpose Verify that the local IRB MACs are learned from L2ALD.

Action On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"

Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"** command.

```
user@PE1> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"

2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x2736568 (adv_entry)
   Advertised metrics:
     Flags: Nexthop Change
     Nexthop: Self
     Localpref: 100
     AS path: [100] I
     Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10 Vector len 4.  Val: 0
       *EVPN  Preference: 170
                Next hop type: Indirect
                Address: 0x26f8354
                Next-hop reference count: 6
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 23:29:08
                Validation State: unverified
                Task: evpna-evpn
                Announcement bits (1): 1-BGP_RT_Background
                AS path: I
                Communities: evpn-default-gateway
                Route Label: 299776
```

Meaning The MAC-only route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB MACs

Purpose Verify that the remote IRB MACs are learned from BGP.

Action On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"

Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

On Router PE2, verify that the remote IRB MACs are learned.

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"** command.

```
user@PE2> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"

2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10/384 (1 entry, 1 announced)
        *BGP  Preference: 170/-101
                Route Distinguisher: 203.0.113.1:100
                Next hop type: Indirect
                Address: 0x26f8d6c
                Next-hop reference count: 10
                Source: 10.255.0.1
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS:   100 Peer AS:   100
                Age: 23:22:17   Metric2: 1
                Validation State: unverified
                Task: BGP_100.10.255.0.1
                Announcement bits (1): 0-evpna-evpn
                AS path: I
                Communities: target:100:100 evpn-default-gateway
                Import Accepted
                Route Label: 299776
                Localpref: 100
                Router ID: 10.255.0.1
                Primary Routing Table bgp.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.255.0.1 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 203.0.113.2 via ge-1/0/8.0
                                Session Id: 0x1
                        10.255.0.1/32 Originating RIB: inet.3
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 203.0.113.2 via ge-1/0/8.0
```

Meaning The MAC-only route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is learned from BGP and tagged with the default gateway extended community.

Verifying Local IRB IPs

Purpose Verify that the local IRB IPs are learned locally by RPD.

Action On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"

Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the **show interfaces irb.0 terse | match inet** command.

```
user@PE1> show interfaces irb.0 terse | match inet

irb.0                   up    up   inet     10.0.0.251/24
```

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"** command.

```
user@PE2> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"

2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x27365a0 (adv_entry)
   Advertised metrics:
     Flags: Nexthop Change
     Nexthop: Self
     Localpref: 100
     AS path: [100] I
     Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251 Vector len 4.  Val:
 0
        *EVPN   Preference: 170 <<<<<
                Next hop type: Indirect
                Address: 0x26f8354
                Next-hop reference count: 6
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 23:48:46
                Validation State: unverified
                Task: evpna-evpn
                Announcement bits (1): 1-BGP_RT_Background
                AS path: I
```

```
                Communities: evpn-default-gateway
                Route Label: 299776
```

Meaning The MAC plus IP route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB IPs

Purpose Verify that the remote IRB IPs are learned from BGP.

Action On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"

Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the **show interfaces irb.0 terse | match inet** command.

```
user@PE1> show interfaces irb.0 terse | match inet

irb.0                   up    up   inet     10.0.0.251/24
```

On Router PE2, verify that the remote IRB IPs are learnt.

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"** command.

```
user@PE2> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"

2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
        *BGP   Preference: 170/-101
                Route Distinguisher: 203.0.113.2:100
                Next hop type: Indirect
                Address: 0x26f8d6c
                Next-hop reference count: 10
                Source: 10.255.0.1
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS:   100 Peer AS:   100
                Age: 23:56:36   Metric2: 1
                Validation State: unverified
                Task: BGP_100.10.255.0.1
                Announcement bits (1): 0-evpna-evpn
```



```
AS path: I
Communities: target:100:100 evpn-default-gateway
Import Accepted
Route Label: 299776
Localpref: 100
Router ID: 10.255.0.1
Primary Routing Table bgp.evpn.0
Indirect next hops: 1
        Protocol next hop: 10.255.0.1 Metric: 1
        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
        Indirect path forwarding next hops: 1
                Next hop type: Router
                Next hop: 203.0.113.2 via ge-1/0/8.0
                Session Id: 0x1
        10.255.0.1/32 Originating RIB: inet.3
          Metric: 1                     Node path count: 1
          Forwarding nexthops: 1
                Nexthop: 203.0.113.2 via ge-1/0/8.0
```

Meaning The MAC plus IP route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is tagged with the default gateway extended community.

Verifying CE-CE Inter-Subnet Forwarding

Purpose Verify inter-subnet forwarding between Routers CE1 and CE2.

Action From operational mode, run the **show route table inet.0** command.

```
user@CE1> show route table inet.0

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

0.0.0.0/0          *[Static/5] 00:15:09
                    > to 10.0.0.251 via ge-1/1/7.0
10.0.0.0/24        *[Direct/0] 1d 23:24:30
                    > via ge-1/1/7.0
10.0.0.1/32        *[Local/0] 1d 23:24:38
                      Local via ge-1/1/7.0
```

From operational mode, run the **ping** command.

```
user@CE1> ping 192.0.2.0 interval 0.1 count 10

PING 192.0.2.0 (192.0.2.0): 56 data bytes
64 bytes from 192.0.2.0: icmp_seq=0 ttl=63 time=0.919 ms
64 bytes from 192.0.2.0: icmp_seq=1 ttl=63 time=0.727 ms
64 bytes from 192.0.2.0: icmp_seq=2 ttl=63 time=0.671 ms
64 bytes from 192.0.2.0: icmp_seq=3 ttl=63 time=0.671 ms
64 bytes from 192.0.2.0: icmp_seq=4 ttl=63 time=0.666 ms
64 bytes from 192.0.2.0: icmp_seq=5 ttl=63 time=0.704 ms
64 bytes from 192.0.2.0: icmp_seq=6 ttl=63 time=0.763 ms
64 bytes from 192.0.2.0: icmp_seq=7 ttl=63 time=0.750 ms
64 bytes from 192.0.2.0: icmp_seq=8 ttl=63 time=12.967 ms
64 bytes from 192.0.2.0: icmp_seq=9 ttl=63 time=0.752 ms

--- 192.0.2.0 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.666/1.959/12.967/3.670 ms
```

Meaning Ping from Router CE1 to Router CE2 is successful.

Verifying CE-PE Inter-Subnet Forwarding

Purpose Verify inter-subnet forwarding between Routers CE1 and PE2.

Action From operational mode, run the **show route table inet.0** command.

```
user@CE1> show route table inet.0

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

0.0.0.0/0          *[Static/5] 00:15:09
                    > to 10.0.0.251 via ge-1/1/7.0
10.0.0.0/24        *[Direct/0] 1d 23:24:30
                    > via ge-1/1/7.0
10.0.0.1/32        *[Local/0] 1d 23:24:38
                      Local via ge-1/1/7.0
```

From operational mode, run the **ping** command.

```
user@CE1> ping 192.0.2.5 interval 0.1 count 10

PING 192.0.2.5 (192.0.2.5): 56 data bytes
64 bytes from 192.0.2.5: icmp_seq=0 ttl=64 time=0.959 ms
64 bytes from 192.0.2.5: icmp_seq=1 ttl=64 time=0.710 ms
64 bytes from 192.0.2.5: icmp_seq=2 ttl=64 time=0.832 ms
64 bytes from 192.0.2.5: icmp_seq=3 ttl=64 time=0.754 ms
64 bytes from 192.0.2.5: icmp_seq=4 ttl=64 time=3.642 ms
64 bytes from 192.0.2.5: icmp_seq=5 ttl=64 time=0.660 ms
64 bytes from 192.0.2.5: icmp_seq=6 ttl=64 time=0.728 ms
64 bytes from 192.0.2.5: icmp_seq=7 ttl=64 time=0.725 ms
64 bytes from 192.0.2.5: icmp_seq=8 ttl=64 time=0.674 ms
64 bytes from 192.0.2.5: icmp_seq=9 ttl=64 time=0.760 ms

--- 192.0.2.5 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.666/1.959/12.967/3.670 ms
```

Meaning Ping from Router CE1 to Router PE2 is successful.

Verifying PE-PE Inter-Subnet Forwarding

Purpose Verify inter-subnet forwarding between Routers PE1 and PE2.

Action From operational mode, run the **show route table vrf.inet.0 192.0.2.5** command.

```
user@PE1> show route table vrf.inet.0 192.0.2.5

vrf.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.5/24       *[EVPN/7] 02:24:32, metric2 1
                     > to 1.0.0.2 via ge-1/0/8.0, Push 299776, Push 299776(top)
```

From operational mode, run the **ping** command.

```
user@CE1> ping routing-instance evpna 192.0.2.5 interval 0.1 count 10

PING 192.0.2.5 (192.0.2.5): 56 data bytes
64 bytes from 192.0.2.5: icmp_seq=0 ttl=64 time=9.613 ms
64 bytes from 192.0.2.5: icmp_seq=1 ttl=64 time=0.789 ms
64 bytes from 192.0.2.5: icmp_seq=2 ttl=64 time=0.803 ms
64 bytes from 192.0.2.5: icmp_seq=3 ttl=64 time=0.695 ms
64 bytes from 192.0.2.5: icmp_seq=4 ttl=64 time=0.742 ms
64 bytes from 192.0.2.5: icmp_seq=5 ttl=64 time=0.702 ms
64 bytes from 192.0.2.5: icmp_seq=6 ttl=64 time=0.725 ms
64 bytes from 192.0.2.5: icmp_seq=7 ttl=64 time=0.730 ms
64 bytes from 192.0.2.5: icmp_seq=8 ttl=64 time=0.713 ms
64 bytes from 192.0.2.5: icmp_seq=9 ttl=64 time=7.370 ms

--- 192.0.2.5 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.695/2.288/9.613/3.142 ms
```

Meaning Ping from Router PE1 to Router PE2's IRB interface is successful.

See Also
- EVPN with IRB Solution Overview on page 600

Example: Configuring an EVPN with IRB Solution on EX9200 Switches

This example shows how to configure an integrated routing and bridging (IRB) solution in an Ethernet VPN (EVPN) deployment.

Requirements

This example uses the following hardware and software components:

- Two EX9200 switches configured as PE routers

- Junos OS Release 14.2 or later running on all the PE routers

Before you begin:

1. Configure the switch interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure LDP.
5. Configure MPLS.

Overview

In an EVPN solution, multiple VLANs can be defined within a particular EVPN instance, and one or more EVPN instances can be associated with a single Layer 3 VPN VRF. In general, each data center tenant is assigned a unique Layer 3 VPN virtual route forwarding (VRF), although the tenant can comprise one or more EVPN instances or VLANs per EVPN instance.

To support this flexibility and scalability factor, the EVPN solution provides support for the IRB interfaces on EX9200 switches to facilitate optimal Layer 2 and Layer 3 forwarding along with virtual machine mobility. The IRB interfaces are configured on each configured VLAN including the default VLAN for an EVPN instance.

IRB is the ability to do Layer 2 switching and Layer 3 routing within a single node, thus avoiding extra hops for inter-subnet traffic. The EVPN IRB solution eliminates the default gateway problem using the gateway MAC and IP synchronization, and avoids the triangular routing problem with Layer 3 interworking by creating IP host routes for virtual machines (VMs) in the tenant VRFs.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

CE1

```
set interfaces ge-1/1/7 vlan-tagging
set interfaces ge-1/1/7 unit 0 vlan-id 10
set interfaces ge-1/1/7 unit 0 family inet address 10.0.0.1/24
set routing-options static route 198.51.100.0/24 next-hop 10.0.0.251
```

PE1

```
set interfaces ge-1/0/8 unit 0 family inet address 192.0.2.1/24
set interfaces ge-1/0/8 unit 0 family mpls
set interfaces ge-1/1/8 flexible-vlan-tagging
set interfaces ge-1/1/8 encapsulation flexible-ethernet-services
set interfaces ge-1/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/8 unit 0 vlan-id 10
set interfaces irb unit 0 family inet address 10.0.0.251/24
```

```
set interfaces lo0 unit 0 family inet address 203.0.113.1/32
set routing-options router-id 203.0.113.1
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 203.0.113.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 203.0.113.2
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 10
set routing-instances evpna interface ge-1/1/8.0
set routing-instances evpna l3-interface irb.0
set routing-instances evpna route-distinguisher 203.0.113.1:100
set routing-instances evpna vrf-target target:100:100
set routing-instances evpna protocols evpn interface ge-1/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 203.0.113.1:300
set routing-instances vrf vrf-target target:100:300
set routing-instances vrf vrf-table-label
```

PE2

```
set interfaces ge-2/0/8 unit 0 family inet address 192.0.2.2/24
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces ge-2/1/8 flexible-vlan-tagging
set interfaces ge-2/1/8 encapsulation flexible-ethernet-services
set interfaces ge-2/1/8 unit 0 encapsulation vlan-bridge
set interfaces ge-2/1/8 unit 0 vlan-id 20
set interfaces irb unit 0 family inet address 198.51.100.251/24
set interfaces lo0 unit 0 family inet address 203.0.113.2/32
set routing-options router-id 203.0.113.2
set routing-options autonomous-system 100
set routing-options forwarding-table chained-composite-next-hop ingress evpn
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 203.0.113.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 203.0.113.1
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances evpna instance-type evpn
set routing-instances evpna vlan-id 20
set routing-instances evpna interface ge-2/1/8.0
set routing-instances evpna l3-interface irb.0
set routing-instances evpna route-distinguisher 203.0.113.2:100
```

```
set routing-instances evpna vrf-target target:200:100
set routing-instances evpna protocols evpn interface ge-2/1/8.0
set routing-instances vrf instance-type vrf
set routing-instances vrf interface irb.0
set routing-instances vrf route-distinguisher 203.0.113.2:300
set routing-instances vrf vrf-target target:200:300
set routing-instances vrf vrf-table-label
```

CE2

```
set interfaces ge-2/1/7 unit 0 vlan-id 20
set interfaces ge-2/1/7 unit 0 family inet address 198.51.100.2/24
set routing-options static route 10.0.0.0/24 next-hop 198.51.100.251
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2, after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Router PE1 interfaces.

```
[edit interfaces]
user@PE1# set ge-1/0/8 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-1/0/8 unit 0 family mpls
user@PE1# set ge-1/1/8 flexible-vlan-tagging
user@PE1# set ge-1/1/8 encapsulation flexible-ethernet-services
user@PE1# set ge-1/1/8 unit 0 encapsulation vlan-bridge
user@PE1# set ge-1/1/8 unit 0 vlan-id 10
user@PE1# set irb unit 0 family inet address 10.0.0.251/24
user@PE1# set lo0 unit 0 family inet address 203.0.113.1/32
```

2. Set the router ID and autonomous system number for Router PE1.

```
[edit routing-options]
user@PE1# set router-id 203.0.113.1
user@PE1# set autonomous-system 100
```

3. Configure the chained composite next hop for EVPN.

```
[edit routing-options]
user@PE1# set forwarding-table chained-composite-next-hop ingress evpn
```

4. Enable LDP on all interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ldp interface all
user@PE1# set ldp interface fxp0.0 disable
```

5. Enable MPLS on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls mpls interface fxp0.0 disable
```

6. Configure the BGP group for Router PE1.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
```

7. Assign local and neighbor addresses to the ibgp BGP group for Router PE1 to peer with Router PE2.

```
[edit protocols]
user@PE1# set bgp group ibgp local-address 203.0.113.1
user@PE1# set bgp group ibgp neighbor 203.0.113.2
```

8. Include the EVPN signaling Network Layer Reachability Information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@PE1# set bgp group ibgp family evpn signaling
```

9. Configure OSPF on all the interfaces of Router PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

10. Configure the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpna instance-type evpn
```

11. Set the VLAN identifier for the bridging domain in the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vlan-id 10
```

12. Configure the interface name for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna interface ge-1/1/8.0
```

13. Configure the IRB interface as the routing interface for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna l3-interface irb.0
```

14. Configure the route distinguisher for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna route-distinguisher 203.0.113.1:100
```

15. Configure the VPN routing and forwarding (VRF) target community for the evpna routing instance.

```
[edit routing-instances]
user@PE1# set evpna vrf-target target:100:100
```

16. Assign the interface name that connects the PE1 site to the VPN.

```
[edit routing-instances]
user@PE1# set evpna protocols evpn interface ge-1/1/8.0
```

17. Configure the VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
```

18. Configure the IRB interface as the routing interface for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf interface irb.0
```

19. Configure the route distinguisher for the vrf routing instancee.

```
[edit routing-instances]
user@PE1# set vrf route-distinguisher 203.0.113.1:300
```

20. Configure the VRF label for the vrf routing instance.

```
[edit routing-instances]
user@PE1# set vrf vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show routing-options**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-1/0/8 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
        family mpls;
    }
}
ge-1/1/8 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
irb {
    unit 0 {
        family inet {
            address 10.0.0.251/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 203.0.113.1/32 {
            }
        }
    }
}
```

```
user@PE1# show routing-options
router-id 203.0.113.1;
autonomous-system 100;
forwarding-table {
    chained-composite-next-hop {
        ingress {
            evpn;
        }
    }
}
```

```
user@PE1# show protocols
```

```
ldp {
   interface all;
   interface fxp0.0 {
      disable;
   }
}
mpls {
   interface all;
   interface fxp0.0 {
      disable;
   }
}
bgp {
   group ibgp {
      type internal;
      local-address 203.0.113.1;
      family evpn {
         signaling;
      }
      neighbor 203.0.113.2;
   }
}
ospf {
   area 0.0.0.0 {
      interface all;
      interface fxp0.0 {
         disable;
      }
   }
}
```

```
user@PE1# show routing-instances
evpna {
   instance-type evpn;
   vlan-id 10;
   interface ge-1/1/8.0;
   l3-interface irb.0;
   route-distinguisher 203.0.113.1:100;
   vrf-target target:100:100;
   protocols {
      evpn {
         interface ge-1/1/8.0;
      }
   }
}
vrf {
   instance-type vrf;
   interface irb.0;
   route-distinguisher 203.0.113.1:300;
   vrf-target target:100:300;
   vrf-table-label;
}
```

Verification

Confirm that the configuration is working properly.

Verifying Local IRB MACs

Purpose Verify that the local IRB MACs are learned from L2ALD.

Action On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"

Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"** command.

```
user@PE1> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"

2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x2736568 (adv_entry)
   Advertised metrics:
     Flags: Nexthop Change
     Nexthop: Self
     Localpref: 100
     AS path: [100] I
     Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10 Vector len 4.  Val: 0
       *EVPN  Preference: 170
                Next hop type: Indirect
                Address: 0x26f8354
                Next-hop reference count: 6
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 23:29:08
                Validation State: unverified
                Task: evpna-evpn
                Announcement bits (1): 1-BGP_RT_Background
                AS path: I
                Communities: evpn-default-gateway
                Route Label: 299776
```

Meaning The MAC-only route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB MACs

Purpose Verify that the remote IRB MACs are learned from BGP.

Action On Router PE1, determine the MAC address of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"

Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

On Router PE2, verify that the remote IRB MACs are learned.

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"** command.

```
user@PE2> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10"

2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10/384 (1 entry, 1 announced)
        *BGP  Preference: 170/-101
                Route Distinguisher: 2.91.223.24:100
                Next hop type: Indirect
                Address: 0x26f8d6c
                Next-hop reference count: 10
                Source: 10.255.0.1
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS:   100 Peer AS:   100
                Age: 23:22:17   Metric2: 1
                Validation State: unverified
                Task: BGP_100.10.255.0.1
                Announcement bits (1): 0-evpna-evpn
                AS path: I
                Communities: target:100:100 evpn-default-gateway
                Import Accepted
                Route Label: 299776
                Localpref: 100
                Router ID: 10.255.0.1
                Primary Routing Table bgp.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.255.0.1 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
```

```
                              Next hop: 1.0.0.1 via ge-1/0/8.0
                              Session Id: 0x1
                        10.255.0.1/32 Originating RIB: inet.3
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 1
                              Nexthop: 1.0.0.1 via ge-1/0/8.0
```

Meaning The MAC-only route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is learned from BGP and tagged with the default gateway extended community.

Verifying Local IRB IPs

Purpose Verify that the local IRB IPs are learned locally by RPD.

Action On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"
```

```
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the **show interfaces irb.0 terse | match inet** command.

```
user@PE1> show interfaces irb.0 terse | match inet
```

```
irb.0                   up    up   inet     10.0.0.251/24
```

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"** command.

```
user@PE2> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"
```

```
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group PE type Internal) Type 1 val 0x27365a0 (adv_entry)
   Advertised metrics:
     Flags: Nexthop Change
     Nexthop: Self
     Localpref: 100
     AS path: [100] I
     Communities: target:100:100 evpn-default-gateway
Path 2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251 Vector len 4.  Val:
 0
        *EVPN   Preference: 170 <<<<<
                Next hop type: Indirect
```

```
                Address: 0x26f8354
                Next-hop reference count: 6
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 23:48:46
                Validation State: unverified
                Task: evpna-evpn
                Announcement bits (1): 1-BGP_RT_Background
                AS path: I
                Communities: evpn-default-gateway
                Route Label: 299776
```

Meaning The MAC plus IP route for the local IRB interface appears in the EVPN instance route table on Router PE1 and is learned from EVPN and tagged with the default gateway extended community.

Verifying Remote IRB IPs

Purpose Verify that the remote IRB IPs are learned from BGP.

Action On Router PE1, determine the MAC and IP addresses of the local IRB interface.

From operational mode, run the **show interfaces irb extensive | match "Current address"** command.

```
user@PE1> show interfaces irb extensive | match "Current address"
```

```
Current address: a8:d0:e5:54:0d:10, Hardware address: a8:d0:e5:54:0d:10
```

From operational mode, run the **show interfaces irb.0 terse | match inet** command.

```
user@PE1> show interfaces irb.0 terse | match inet
```

```
irb.0                   up    up   inet     10.0.0.251/24
```

On Router PE2, verify that the remote IRB IPs are learnt.

From operational mode, run the **show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"** command.

```
user@PE2> show route table evpna.evpn.0 extensive | find "a8:d0:e5:54:0d:10::10.0.0.251"
```

```
2:10.255.0.1:100::0::100::a8:d0:e5:54:0d:10::10.0.0.251/384 (1 entry, 1 announced)
        *BGP   Preference: 170/-101
                Route Distinguisher: 2.91.223.216:100
                Next hop type: Indirect
                Address: 0x26f8d6c
```

```
                Next-hop reference count: 10
                Source: 10.255.0.1
                Protocol next hop: 10.255.0.1
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS:   100 Peer AS:   100
                Age: 23:56:36   Metric2: 1
                Validation State: unverified
                Task: BGP_100.10.255.0.1
                Announcement bits (1): 0-evpna-evpn
                AS path: I
                Communities: target:100:100 evpn-default-gateway
                Import Accepted
                Route Label: 299776
                Localpref: 100
                Router ID: 10.255.0.1
                Primary Routing Table bgp.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.255.0.1 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 1.0.0.1 via ge-1/0/8.0
                                Session Id: 0x1
                        10.255.0.1/32 Originating RIB: inet.3
                          Metric: 1                     Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 1.0.0.1 via ge-1/0/8.0
```

Meaning The MAC plus IP route for the remote IRB interface appears in the EVPN instance route table on Router PE2 and is tagged with the default gateway extended community.

Verifying CE-CE Inter-Subnet Forwarding

Purpose Verify inter-subnet forwarding between Routers CE1 and CE2.

Action From operational mode, run the **show route table inet.0** command.

```
user@CE1> show route table inet.0

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

0.0.0.0/0          *[Static/5] 00:15:09
                    > to 10.0.0.251 via ge-1/1/7.0
10.0.0.0/24        *[Direct/0] 1d 23:24:30
                    > via ge-1/1/7.0
10.0.0.1/32        *[Local/0] 1d 23:24:38
                      Local via ge-1/1/7.0
```

From operational mode, run the **ping** command.

```
user@CE1> ping 198.51.100.2 interval 0.1 count 10

PING 198.51.100.2 (20.0.0.2): 56 data bytes
64 bytes from 198.51.100.2: icmp_seq=0 ttl=63 time=0.919 ms
64 bytes from 198.51.100.2: icmp_seq=1 ttl=63 time=0.727 ms
64 bytes from 198.51.100.2: icmp_seq=2 ttl=63 time=0.671 ms
64 bytes from 198.51.100.2: icmp_seq=3 ttl=63 time=0.671 ms
64 bytes from 198.51.100.2: icmp_seq=4 ttl=63 time=0.666 ms
64 bytes from 198.51.100.2: icmp_seq=5 ttl=63 time=0.704 ms
64 bytes from 198.51.100.2: icmp_seq=6 ttl=63 time=0.763 ms
64 bytes from 198.51.100.2: icmp_seq=7 ttl=63 time=0.750 ms
64 bytes from 198.51.100.2: icmp_seq=8 ttl=63 time=12.967 ms
64 bytes from 198.51.100.2: icmp_seq=9 ttl=63 time=0.752 ms

--- 198.51.100.2 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.666/1.959/12.967/3.670 ms
```

Meaning Ping from Router CE1 to Router CE2 is successful.

Related Documentation

- An EVPN with IRB Solution on EX9200 Switches Overview on page 605

CHAPTER 21

Configuring IGMP Snooping with EVPN-MPLS

Overview of Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment

Starting with Junos OS Release 18.2R1, MX Series routers with MPCs, vMX routers, and EX9200 switches support multicast traffic forwarding with Internet Group Management Protocol (IGMP) snooping in an Ethernet VPN (EVPN) over MPLS environment. Multicast source and receiver hosts in the EVPN instance (EVI) can each be single-homed to one provider edge (PE) device or multihomed in all-active mode to multiple provider edge (PE) devices, and can be attached to PE devices at the same site or at different sites. For receivers that are multihomed to multiple PE devices, IGMP state information is synchronized among the peer PE devices. IGMP snooping can be enabled for multiple EVIs, and either for specific bridge domains or VLANs in an EVI or all bridge domains or VLANs within an EVPN virtual switch instance. Multicast traffic can be forwarded within a bridge domain or VLAN, and can be routed across bridge domains or VLANs at Layer 3 using IRB interfaces.

NOTE: This environment is supported using IGMPv2 only.

Benefits of Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment

- In an environment with significant multicast traffic, using IGMP snooping constrains multicast traffic in a broadcast domain or VLAN to interested receivers and multicast devices, which conserves network bandwidth.
- Synchronizing IGMP state among all EVPN PE devices for multihomed receivers ensures that all subscribed listeners receive multicast traffic, even in cases such as the following:
  - IGMP join and leave messages for a multicast group might arrive on a PE device that is not the Ethernet segment's designated forwarder (DF).
  - An IGMP leave message for a multicast group arrives at a different PE device than the PE device where the IGMP join message for the multicast group was received.

Multicast Forwarding with IGMP Snooping in Single-homed or Multihomed Ethernet Segments

For redundancy, an EVPN-MPLS environment can have multicast sources and receivers multihomed to a set of peer PE devices that are in all-active mode. When all PE devices in the EVI have IGMP snooping enabled in proxy mode, the IGMP state is synchronized among the multihomed peer PE devices, and multicast traffic can reach all listeners.

IGMP Join and Leave Route Synchronization Among Multihoming Peer PE Devices

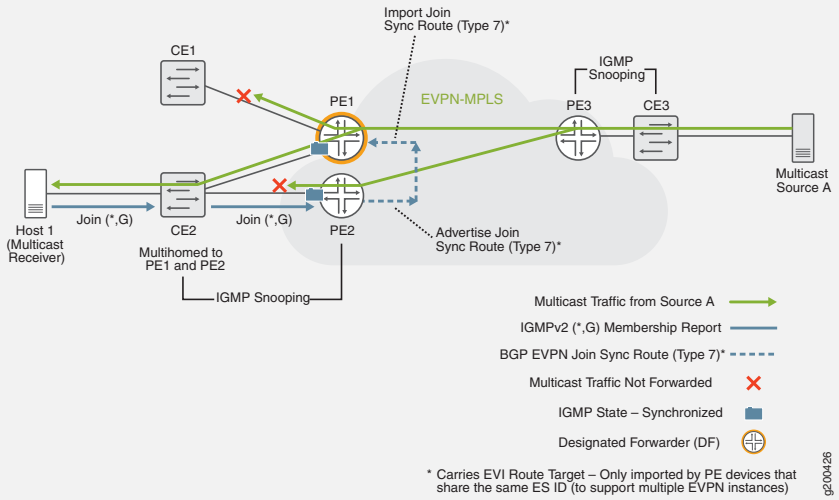
In an EVI with receivers that are multihomed to multiple PE devices, corresponding IGMP join and leave messages for multicast group management might not be sent to the same PE device, so IGMP synchronization is required among all the PE devices to share IGMP state. PE devices with IGMP snooping enabled in this environment exchange BGP EVPN Type 7 (IGMP Join Sync Route) and Type 8 (IGMP Leave Synch Route) network layer reachability information (NLRI) to synchronize IGMP join and leave reports received on multihomed interfaces. IGMP snooping with multihoming in this environment is supported only in all-active mode.

The advertised EVPN Type 7 and Type 8 routes also carry EVI route target extended community attributes associated with multihomed EVIs, so multiple EVPN routing instances can be supported simultaneously. These routes are only imported by PE devices that share the same Ethernet segment ID.

EVPN Type 7 and Type 8 route advertisement is not needed for hosts that are single-homed. However, IGMP snooping must also be configured in this case to enable selective multicast forwarding on the access side, and thus bandwidth is saved by only forwarding multicast traffic to interested receivers.

Figure 61 on page 653 shows a multicast source behind customer edge device CE3, and a multicast receiver Host 1 behind CE2 that is multihomed to PE devices PE1 and PE2.

Figure 61: Multicast Forwarding with IGMP Snooping for Multihomed Receivers in EVPN-MPLS



In Figure 61 on page 653, PE2 receives the join message, and advertises the EVPN Type 7 Join Sync Route to the EVPN core. PE1 imports the Type 7 route to synchronize the IGMP state among the PE devices to which the receiver is multihomed.

Multicast Traffic Forwarding with Single-homed or Multihomed Receivers

With IGMP snooping enabled, PE devices perform inclusive multicast forwarding using ingress replication to forward multicast traffic into the EVPN core, and selective multicast forwarding on the access side, as follows:

- To ensure multicast traffic reaches all remote PE devices, the ingress PE device for a multicast source floods the multicast traffic into the EVPN core to reach all remote PE devices using Inclusive Multicast Ethernet Tag (IMET) routing with ingress replication.
- If a multihoming PE device receives multicast traffic from the EVPN core and it is the DF for an interested receiver for the multicast group, the PE device forwards the traffic.
- If a multihoming PE device receives multicast traffic from the EVPN core and it is not the DF for any interested receivers, the PE device does not forward the traffic.
- On the access side, upon receiving multicast data from the EVPN core, PE devices perform selective multicast forwarding only to interested receivers. Single-homing PE devices use learned IGMP snooping information, and multihoming PE devices use both IGMP snooping information and EVPN Type 7 routes.

For example, in Figure 61 on page 653, PE3 is the ingress PE device for Multicast Source A and forwards the multicast traffic to the EVPN core. Host 1 is a multicast receiver that is multihomed to PE1 and PE2 by way of customer edge device CE2. PE1 and PE2 have synchronized IGMP state information about receiver Host 1, and both PE devices receive the multicast traffic from the EVPN core. Because PE1 is currently the elected DF for Host 1, PE1 forwards the data to Host 1, but based on the IGMP snooping information and

imported EVPN Type 7 routing information, PE1 does not forward the data to CE1. PE2 is not the DF, so PE2 does not forward the traffic toward Host 1.

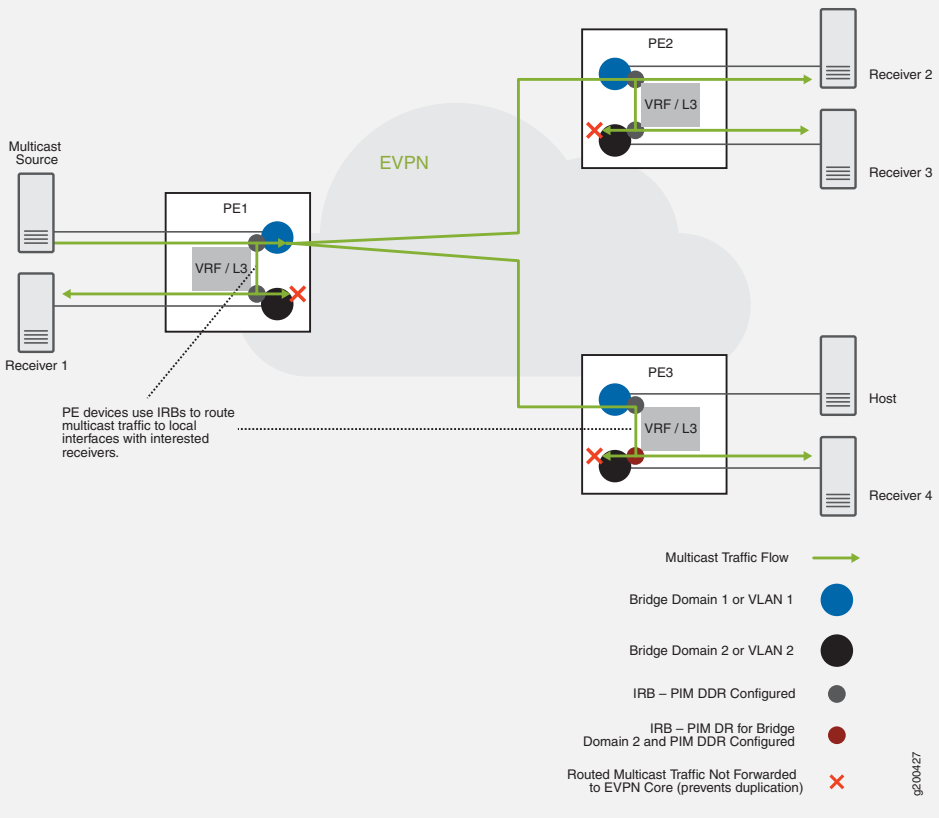
IGMP Snooping with Multicast Forwarding Between Bridge Domains or VLANs at Layer 3

For multicast forwarding between bridge domains or VLANs in this environment, PE devices use Protocol Independent Multicast (PIM) in distributed designated router (DDR) mode on IRB interfaces.

The IRB interfaces on PE devices route multicast traffic between bridge domains or VLANs as follows:

- Upon receiving multicast traffic on an IRB interface from a multicast source, the PE device routes the traffic to any IRBs that have PIM enabled and are configured for bridge domains or VLANs with interested local receivers for the multicast group.
- With PIM DDR configured, PE devices with IRB interfaces route multicast traffic to local receivers whether or not the IRB is the elected PIM designated router (DR).
- To prevent multicast traffic duplication, IRB routed multicast traffic is not forwarded back to the EVPN core.

Figure 62: Routing Multicast Traffic Between Bridge Domains (VLANs) with IRBs in EVPN-MPLS



For example, Figure 62 on page 654 shows an EVPN instance has three PE devices, PE1, PE2, and PE3, with IRB interfaces configured on each PE device for two bridge domains or VLANs. The multicast source on bridge domain 1 (VLAN 1) sends multicast traffic to PE1, and PE1 uses inclusive multicast forwarding with ingress replication to forward the traffic to the EVPN core to reach other PE devices with interested receivers. PE1 also routes the traffic to interested receivers on bridge domain 2 (VLAN 2) locally through the IRB interfaces, even though PE3 is the PIM DR for bridge domain 2 (VLAN 2). The IRB interfaces on all PE devices that route traffic to receivers on bridge domain 2 (VLAN 2) do not forward the routed traffic back out into the EVPN core. PE2 receives the multicast traffic from the EVPN core and forwards or locally routes the traffic on each bridge domain (VLAN) that has interested receivers. PE3 routes the traffic locally to bridge domain 2 only, because there are no interested receivers on bridge domain 1 (VLAN 1).

Requirements and Limitations of Multicast with IGMP Snooping in an EVPN-MPLS Environment

This environment requires the following:

- IGMP snooping is enabled in proxy mode.
- IGMP snooping is supported with single-homing and all-active multihoming modes only.
- Only IGMPv2 is supported with IGMP snooping.
- All bridge domains or VLANs with multicast sources and receivers are configured on all PE devices in the EVI.
- IRB interfaces for routing multicast traffic between bridge domains or VLANs must be configured on all PE devices in the EVI for multicast traffic to be routed locally on each PE device.

  NOTE: **If the bridge domain (VLAN) and an IRB interface are configured on all PE devices, but an IRB interface is down on a PE device, traffic routing between bridge domains (VLANs) for receivers served by that IRB will be impacted.**

- For Layer 3 routing of multicast traffic between bridge domains or VLANs, PIM DDR mode must be configured on all participating IRB interfaces.
- All multihomed peer PE devices support BGP EVPN Type 7 Join Sync Routes and Type 8 Leave Sync Routes.
- PE devices are configured in enhanced IP mode on MX Series routers.

This environment has the following limitations:

- Using PIM passive mode on IRB interfaces in the EVPN instance is not supported.
- IGMP snooping in this environment with an external multicast router instead of IRB interfaces is not supported.
- Receiving or forwarding multicast traffic from or to devices outside of this environment using a PIM gateway is not supported.

- Selective multicast forwarding (advertising EVPN Type 6 Selective Multicast Ethernet Tag (SMET) routes for forwarding only to interested receivers) in the EVPN core is not supported.
- IGMP snooping is not supported with point-to-multipoint (P2MP) multipoint LDP/RSVP replication; only ingress replication is supported.

Release History Table

Release	Description
18.2R1	Starting with Junos OS Release 18.2R1, MX Series routers with MPCs, vMX routers, and EX9200 switches support multicast traffic forwarding with Internet Group Management Protocol (IGMP) snooping in an Ethernet VPN (EVPN) over MPLS environment.

Related Documentation

- Configuring Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment on page 656
- EVPN Multihoming Overview on page 29

Configuring Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment

This topic shows how to configure multicast forwarding with IGMP snooping on MX Series routers with MPCs, vMX routers, and EX9200 switches serving as provider edge (PE) devices in an Ethernet VPN (EVPN) over MPLS environment. Starting in Junos OS Release 18.2R1, statements in the **[edit protocols igmp-snooping]** configuration hierarchy can be configured in this environment on PE devices to enable multicast traffic forwarding for receivers that are multihomed to EVPN provider edge (PE) devices with all-active multihoming.

In this environment, IGMP snooping is enabled in proxy mode. The IGMP snooping configuration is not automatically synchronized among all the multihoming peer PEs, so the same configuration must be committed on each multihomed peer PE device.

In addition, to handle Layer 3 multicast forwarding, PEs with hosts in multicast groups that span bridge domains or VLANs use PIM distributed designated router (PIM DDR) mode to route multicast traffic between bridge domains or VLANs through IRB interfaces. With PIM DDR configured, a PE device routes the traffic to any local IRBs configured for bridge domains or VLANs in the multicast group with interested receivers, even if the IRB is not the elected PIM designated router (DR).

IGMP snooping and multicast forwarding operation ensures that multicast traffic reaches all subscribed receivers within and between bridge domains or VLANs, and preserves bandwidth on the access side by reducing the amount of multicast control and data traffic being forwarded.

This topic describes configuration tasks to set up IGMP snooping in proxy mode, and configure PIM for routing between bridge domains or VLANs on multihoming EVPN PE

devices. It also summarizes the CLI commands that can be used to verify IGMP snooping and multicast forwarding operation in this environment.

Configuring IGMP Snooping for an EVPN or Virtual Switch Routing Instance

- To configure IGMP snooping in proxy mode on a PE device for a routing instance with **instance-type evpn** (for any or all bridge domains or VLANs):

  `user@device#` **set routing-instances *routing-instance-name* protocols igmp-snooping proxy**

  For example, the following configuration includes enabling IGMP snooping in proxy mode for EVPN routing instance evpn-A configured as **instance-type evpn**:

```
routing-instances {
    evpn-A {
        instance-type evpn;
        interface ge-0/0/2.600;
        route-distinguisher 10.255.255.1:100;
        vrf-target target:100:100;
        protocols {
            evpn {
                label-allocation per-instance;
            }
            igmp-snooping proxy;
        }
    }
}
```

- To configure IGMP snooping on a PE device for specific bridge domains or VLANs for an EVPN instance with **instance-type virtual-switch**:

  `user@device#` **set routing-instances *routing-instance-name* bridge-domain *bridge-domain-name* protocols igmp-snooping proxy**

  For example, the following configuration includes enabling IGMP snooping in proxy mode for bridge domains V200, V201, and V202 in EVPN routing instance EVPN-2, which is configured as **instance-type virtual-switch**:

```
routing-instances {
    ...
    EVPN-2 {
        instance-type virtual-switch;
        route-distinguisher 90.90.90.10:2;
        vrf-target target:65530:2;
        protocols {
            evpn {
                encapsulation mpls;
```

```
                extended-vlan-list 200-202;
                default-gateway advertise;
            }
        }
        bridge-domains {
            V200 {
                domain-type bridge;
                vlan-id 200;
                interface ae3.200;
                routing-interface irb.200;
                protocols {
                    igmp-snooping proxy;
                }
            }
            V201 {
                domain-type bridge;
                vlan-id 201;
                interface ae3.201;
                routing-interface irb.201;
                protocols {
                    igmp-snooping proxy;
                }
            }
            V202 {
                domain-type bridge;
                vlan-id 202;
                interface ae3.202;
                routing-interface irb.202;
                protocols {
                    igmp-snooping proxy;
                }
            }
        }
    }
 ...
}
```

Configuring Multicast Forwarding Across Bridge Domains or VLANs with PIM in EVPN-MPLS

PIM must be enabled in distributed DR mode on all IRB interfaces used in an EVPN instance to reach multicast source or receiver hosts for one or more bridge domains or VLANs in the EVPN instance.

- To configure PIM DDR mode to route multicast traffic between bridge domains or VLANs through an IRB interface in an EVPN routing instance:

  `user@device#` **set routing-instances *routing-instance-name* protocols pim interface *irb-interface-name* distributed-dr**

  For example, the following configuration includes enabling PIM DDR mode on interfaces irb.200, irb.201, and irb.202 in the VRF routing instance IPVPN-2:

```
routing-instances {
    ...
```

```
IPVPN-2 {
    instance-type vrf;
    interface irb.200;
    interface irb.201;
    interface irb.202;
    interface lo0.2;
    route-distinguisher 90.90.90.10:222;
    vrf-target target:65530:2;
    vrf-table-label;
    protocols {
        pim {
            rp {
                local {
                    address 10.88.88.10;
                }
            }
            interface irb.200 {
                distributed-dr;
            }
            interface irb.201 {
                distributed-dr;
            }
            interface irb.202 {
                distributed-dr;
            }
        }
    }
}
```

Viewing IGMP Snooping Multicast Information for EVPN-MPLS in the CLI

The following EVPN commands are supported to view IGMP snooping multicast information in an EVPN-MPLS environment. The output of these commands includes information learned from native IGMP snooping on a PE device and learned from EVPN Type 7 Join Sync Route and Type 8 Leave Sync Route messages.

- **show evpn igmp-snooping database**
- **show evpn multicast-snooping next-hops**
- **show igmp snooping evpn database**
- **show igmp snooping evpn membership**

The following CLI commands are supported to view the native IGMP snooping information on a PE. The output of these commands will not include information learned from the exchange of EVPN Type 7 and Type 8 IGMP messages.

- **show igmp snooping interface**
- **show igmp snooping membership**
- **show igmp snooping statistics**

The following commands can be used to view EVPN Type 7 IGMP Join Sync Routes or Type 8 Leave Sync Routes in BGP and EVPN routing tables:

- show route table **bgp.evpn.0 match-prefix 7***

  show route table __default_evpn__.evpn.0 match-prefix 7:*

  show route table __default_evpn__.evpn.0 match-prefix 7:* protocol evpn

  show route table __default_evpn__.evpn.0 match-prefix 7:* protocol bgp

- **show route table bgp.evpn.0 match-prefix 8:***

  show route table __default_evpn__.evpn.0 match-prefix 8:*

  show route table __default_evpn__.evpn.0 match-prefix 8:* protocol evpn

  show route table __default_evpn__.evpn.0 match-prefix 8:* protocol bgp

For example, to view EVPN Type 7 IGMP routes in the **__default_evpn__.evpn.0** routing table, use the following command:

```
user@host> show route table __default_evpn__.evpn.0 protocol evpn match-prefix 7:* extensive

Sep 21 02:25:12
__default_evpn__.evpn.0: 32 destinations, 32 routes (32 active, 0 holddown, 0
hidden)
7:90.90.90.10:1::11111111111111111::100::0.0.0.0::233.252.0.1::90.90.90.10/600
(1 entry, 1 announced)
TSI:
Page 0 idx 0, (group IBGP type Internal) Type 1 val 0xb23c68c (adv_entry)
   Advertised metrics:
     Nexthop: 90.90.90.10
     Localpref: 100
     AS path: [65530] I
     Communities: es-import-target:11-11-11-11-11-11 evi-rt:65530:1 Path
7:90.90.90.10:1::11111111111111111::100::0.0.0.0::233.252.0.1::90.90.90.10 Vector
 len 4.  Val: 0
        *EVPN   Preference: 170
                Next hop type: Indirect, Next hop index: 0
                Address: 0xb2e5e10
                Next-hop reference count: 61
                Protocol next hop: 90.90.90.10
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 2:12:30
                Validation State: unverified
                Task: __default_evpn__-evpn
                Announcement bits (1): 1-BGP_RT_Background
                AS path: I
                Communities: es-import-target:11-11-11-11-11-11 evi-rt:65530:1
                IGMP flags: 0x2

7:90.90.90.10:3::33333333333333333::100::0.0.0.0::233.252.0.1::90.90.90.10/600
(1 entry, 1 announced)
TSI:
Page 0 idx 0, (group IBGP type Internal) Type 1 val 0xb23c6a8 (adv_entry)
   Advertised metrics:
     Nexthop: 90.90.90.10
     Localpref: 100
```

```
    AS path: [65530] I
    Communities: es-import-target:33-33-33-33-33-33 evi-rt:65530:3 Path
7:90.90.90.10:3::333333333333333333::100::0.0.0.0::233.252.0.1::90.90.90.10 Vector
 len 4.  Val: 0
       *EVPN   Preference: 170
               Next hop type: Indirect, Next hop index: 0
               Address: 0xb2e5e10
               Next-hop reference count: 61
               Protocol next hop: 90.90.90.10
               Indirect next hop: 0x0 - INH Session ID: 0x0
               State: <Active Int Ext>
               Age: 2:12:30
               Validation State: unverified
               Task: __default_evpn__-evpn
               Announcement bits (1): 1-BGP_RT_Background
               AS path: I
               Communities: es-import-target:33-33-33-33-33-33 evi-rt:65530:3
               IGMP flags: 0x2
```

Related Documentation

- Overview of Multicast Forwarding with IGMP Snooping in an EVPN-MPLS Environment on page 651

PART 4

EVPN-VPWS

CHAPTER 22

Configuring VPWS Service with EVPN Mechanisms

Overview of VPWS with EVPN Signaling Mechanisms

An Ethernet VPN (EVPN) enables you to connect dispersed customer sites using a Layer 2 virtual bridge. As compared with other types of Layer 2 VPNs, an EVPN consists of customer edge (CE) devices (host, router, or switch) connected to provider edge (PE) routers. The PE routers can include an MPLS edge switch (MES) that acts at the edge of the MPLS infrastructure. Either an MX Series 5G Universal Routing Platform or a standalone switch can be configured to act as an MES. You can deploy multiple EVPNs within a service provider network, each providing network connectivity to a customer while ensuring that the traffic sharing on that network remains private.

Virtual private wire service (VPWS) Layer 2 VPNs employ Layer 2 services over MPLS to build a topology of point-to-point connections that connect end customer sites in a VPN. The service provisioned with these Layer 2 VPNs is known as VPWS. You can configure a VPWS instance on each associated edge device for each VPWS Layer 2 VPN.

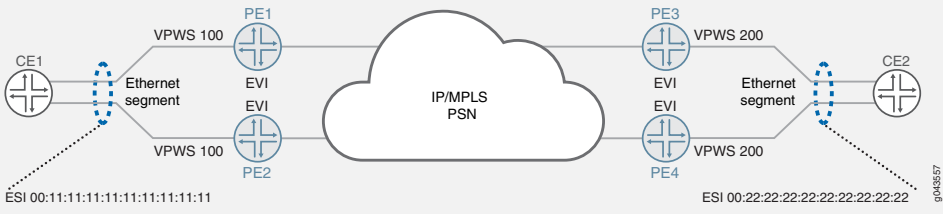
An EVPN-VPWS network provides a framework for delivering VPWS with EVPN signaling mechanisms. The advantages of VPWS with EVPN mechanisms are single-active or all-active multihoming capabilities and support for Inter-autonomous system (AS) options associated with BGP-signaled VPNs. Metro Ethernet Forum (MEF) describes the following two service models for VPWS:

- Ethernet private line (EPL)— EPL provides a point-to-point Ethernet virtual connection (EVC) between a pair of dedicated user–network interfaces (UNIs) that is between a pair of Ethernet segment identifiers (ESIs) with a high degree of transparency.
- Ethernet virtual private line (EVPL)— EVPL provides point-to-point EVC between {ESI, VLAN} pairs. EVPL allows service multiplexing; that is multiple EVCs or Ethernet services per UNI.

An EVPN-VPWS network is supported on an EVPN-MPLS network.

"EVPN-VPWS" on page 663 illustrates an EVPN-VPWS network. Device CE1 is multihomed to Routers PE1 and PE2 and Device CE2 is multihomed to Routers PE3 and PE4. Device CE2 has two potential paths to reach CE1, and depending on the multihoming mode of redundancy, only one path or all paths can be active at any one time. When a CE device is multihomed to two or more PE routers, the set of Ethernet links constitutes an Ethernet segment. An Ethernet segment appears as a link aggregation group (LAG) to the CE device. The links from PE1 and PE2 to CE1 and PE3 and PE4 to CE2 form an Ethernet segment.

Figure 63: VPWS in EVPN



An Ethernet segment must have a unique nonzero identifier, called the *Ethernet segment identifier (ESI)*. The ESI is encoded as a 10-octet integer. When manually configuring an ESI value, the most significant octet, known as the *type byte*, must be 00. When a single-homed CE device is attached to an Ethernet segment, the entire ESI value is zero. The Ethernet segment of the multihomed Device CE1 has an ESI value of 00:11:11:11:11:11:11:11:11:11 and the Ethernet segment of the multihomed Device CE2 has an ESI value of 00:22:22:22:22:22:22:22:22:22.

An EVPN instance (EVI) is an EVPN routing and forwarding instance spanning across all the PE routers participating in that VPN. An EVI is configured on the PE routers on a per-customer basis. Each EVI has a unique route distinguisher and one or more route targets. An EVI is configured on PE1, PE2, PE3, and PE4. An Ethernet tag identifies a particular broadcast domain, such as a VLAN. An EVI consists of one or more broadcast domains. Ethernet tags are assigned to the broadcast domains of a given EVI by the provider of that EVPN. Each PE router in that EVI performs a mapping between broadcast domain identifiers understood by each of its attached CE devices and the corresponding Ethernet tag. Depending on the multihoming mode of redundancy, only one path or all paths can be active at any one time.

The multihoming mode of operation along with VPWS service identifiers determine which PE router or routers forward and receive traffic in the EVPN-VPWS network. The VPWS service identifier identifies the endpoints of the EVPN-VPWS network. These endpoints are autodiscovered by BGP and are used to exchange the service labels(learned from the respective PE routers) that are used by autodiscovered routes per EVI route type. The service identifier is of two types:

- Local— Unique local VPWS service identifier. This is a logical identifier mapped to a physical interface of a PE router connected to the customer site that is forwarding the traffic to a remote VPWS service identifier.
- Remote—Unique remote VPWS service identifier. This is a logical identifier mapped to a physical interface of a PE router connected to the customer site that is receiving the traffic forwarded from the local VPWS service identifier.

The PE router forwarding traffic to the CE device uses MPLS LSP to forward traffic. If a failure occurs over this path, a new designated forwarder is elected to forward the traffic to the CE device.

The EVPN-VPWS network uses only an autodiscovered route per ESI and an autodiscovered router per EVI route types. In autodiscovered routes per EVI, the 24-bit values of the Ethernet tag are encoded with the VPWS service identifier. The autodiscovered route per ESI is encoded with the route targets of all the EVPN-VPWS instances connected to the ESI on the advertising PE router . When the PE router loses its connectivity to the ESI, it withdraws the autodiscovered route per ESI, resulting in faster convergence. The receiving PE router updates the forwarding next hop of the VPWS service identifier impacted by the failure. Depending on the mode of operation, these two endpoints of the EVPN-VPWS network can be colocated on the same PE router or on different PE routers. The different modes of operation in an EVPN-VPWS network are as follows:

- Local switching—In this mode, the VPWS endpoints (that is, local and remote service identifiers) are connected through the local interfaces configured on the same PE router. Traffic from one CE router is forwarded to another CE router through the same PE router.
- Single-homing–When a PE router is connected to a single-homed customer site, this mode is in operation.
- Active-standby multihoming—In this mode, only a single PE router among a group of PE routers with the same VPWS service identifier attached to an Ethernet segment is allowed to forward traffic to and from that Ethernet segment. The process of electing one among many PE routers with the same VPWS service identifer is known as the *designated forwarder (DF) election*. Each PE router connected to the Ethernet segment with the same VPWS service identifier participates in the DF election and informs the network of its status. The status can be as follows:
  - Designated forwarder(DF)—This is the designated forwarder for forwarding the current traffic.
  - Backup designated forwarder(BDF)—This becomes the DF in case the current DF encounters a failure.
  - Non-designated forwarder(non-DF)—This is neither the DF nor the BDF. When more than two PE routers are part of an ESI redundancy group, then this PE router becomes a non-DF.

  To configure the active-standby mode, include the ESI value and the **single-active** statement under the [**edit interfaces**] hierarchy level. Configure the **local** and the **remote** VPWS service identifier under the [**edit routing-instances** ***vpws-routing-instance*** **protocols evpn interface** ***interface-name*** **vpws-service-id**] for each PE router connected to the Ethernet segment.

  In "EVPN-VPWS" on page 663, for the PE routers connected to Device CE1, Router PE1 is assumed to be the DF and PE2 is assumed to be the BDF for VPWS service identifier 100. For the PE routers connected to CE2, PE3 is assumed to be the DF and Router PE4 is assumed to be the BDF for VPWS service identifier 200. PE2 and PE4 indicate their BDF status to CE1 and CE2 by setting their circuit cross-connect (CCC)-Down flag on their respective interfaces.
- Active-active multi-homing–In active-active multi homing, the CE device is connected to the PE routers with the same local VPWS identifier through the LAG interface so that the traffic is load-balanced among the set of multihomed PE routers with the same remote VPWS service identifiers. Here, election of DF is not required, because all the PE routers connected to the LAG interface participate in forwarding the traffic. In "EVPN-VPWS" on page 663, Device CE1 forwards traffic to PE1 and PE2 with VPWS service identifier 100 and CE2 forwards traffic to PE3 and PE4 with VPWS service identifier 200. PE1 forwards the traffic to PE3 and PE4 with remote VPWS service identifier 200. PE2 forwards the traffic to PE3 and PE4 . Similarly, traffic from CE2 to PE3 and PE4 with VPWS service identifier is load-balanced across PE routers with VPWS service identifier 100 connected to CE1.

NSR and Unified ISSU Support on VPWS with EVPN

Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover. When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets. Unified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU.

To enable GRES, include the **graceful-switchover** statement at the **[edit chassis redundancy]** hierarchy level.

To enable NSR, include the **nonstop-routing** statement at the **[edit routing-options]** hierarchy level and the **commit synchronize** statement at the **[edit system]** hierarchy level.

Related Documentation

- EVPN Multihoming Overview on page 29
- Configuring VPWS with EVPN Signaling Mechanisms on page 681
- Example: Configuring VPWS with EVPN Signaling Mechanisms on page 683
- **vpws-service-id on page 1090**
- **show evpn vpws-instance on page 1193**

Overview of Flexible Cross-Connect Support on VPWS with EVPN

Ethernet VPN virtual private wire service (EVPN-VPWS) delivers the point-to-point services between a pair of Attachment Circuits (ACs). An AC is an attachment circuit or access interface participating in a EVPN E-LINE service. Multi-homing and fast convergence capabilities are also provided by EVPN-VPWS. You can now multiplex a large number of ACs across several physical interfaces onto a single VPWS service tunnel.

The EVPN-VPWS flexible cross-connect (FXC) is introduced to address label resource issue that occurs on some low end access routers (also known as A-leaf) by having a group of ACs under the same EVPN instance (EVI) share the same label. FXC is used to establish the point-to-point Ethernet services.

NOTE: **The MPLS label resource issue does not apply to the PE device (also known as service edge router) or vMX (or MX), that uses pseudowire subscriber logical interface.**

The control plane signaling is based on the exchange of EVPN per EVI auto-discovery (AD) routes between the pair of PEs, for FXC. It is a mandatory requirement that all the point-to-point Ethernet services under the same EVI is uniquely identified by either a single VLAN tag or double VLAN tags, and to ensure the uniqueness, VLAN normalization must be performed at the ingress. This mandatory requirement is applicable on both PE devices for both VLAN-unaware and VLAN-aware services.

The forwarding plane on the router that has limited label resource (access router), the MPLS label is used to identify the EVI. On both access router and PE device, the VLAN(s)

carried in the data packet is used as the de-multiplexer to uniquely identify each local AC for the point-to-point service.

When the access device is a MX router, on the PE device, only single-homing and single-active multi-homing mode is supported. The customer edge (CE) device on the access side is either single-homed to an access router or multi-homed to a set of access routers in a single-active or all-active mode.

Benefits of Pseudowire Headend Termination (PWHT) with EVPN-VPWS FXC Pseudowires

- EVPN-VPWS FXC is an extension to basic EVPN-VPWS PWHT which bundles VLANs from multiple physical ports on access node into *single bundled* EVPN-VPWS pseudowire. The same pseudowire also shares more than one E-LINE service.
- Supports all true VLAN-aware bundle services for the VLAN signaled FXC.
- The FXC gives the benefit by the efficient usage of MPLS label resources. As a result, less labels and pseudowires are required, because of pseudowire bundling. This is useful on low-end access devices to conserve MPLS labels.

FXC on the PE Device (Service Edge Router)

EVPN-VPWS does not signal the VLAN in the control plane. At the PE device, label is allocated per pseudowire subscriber transport logical interface. The PE device interops with the access router that use different label allocation scheme. Encapsulation type **ethernet-ccc** is used on pseudowire subscriber transport logical interface.

Similar to the VLAN bundle service provided by the pseudowire subscriber logical interface at the PE device, for FXC, a group of ACs, each is represented by its unique normalized VLAN(s), is bundled together.

Single Home Support

All ACs at the access router are connected to the single homed end devices under the same EVI are multiplexed into a single point-to-point service tunnel. This bundle shares the same service instance ID and MPLS service label. Only one EVPN per EVI AD route is advertised for this bundle of ACs.

NOTE: The EVPN per EVI AD route advertised for the bundle of local ACs do not get withdrawal until all the ACs go down or until the EVI itself if deactivated or deleted.

To terminate one particular group of ACs at the access router, a separate pseudowire subscriber logical interface must be configured with the corresponding service instance ID at the PE device. The pseudowire subscriber logical interface works in a VLAN bundle mode.

Multi-home Support

In multi-homing of FXC, a separate per EVI AD route is advertised for each multi-home Ethernet segment. The same Ethernet segment identifier (ESI) ACs are bundled together in the same group. This is because the same PE may play the designated forwarder (DF) and non-designated forwarder (NDF) role independently for different Ethernet segment, but the same PE always acts as the DF or NDF for the ACs on the same Ethernet segment at the same time. As pseudowire subscriber logical interface configured with the corresponding service instance ID is used to terminate point-to-point Ethernet segment for the bundle of ACs, different pseudowire subscriber logical interface has to be used to terminate the different group of multi-homed ACs.

To support interoperability with VLAN-unaware multi-homing FXC at the access router, summary is as follows:

- ACs on the same Ethernet segment are bundled to use the same local service instance ID on the access router.
- Each AC group is assigned a unique local service instance ID, when there are many AC groups on the access router.
- At the PE device, a separate pseudowire subscriber logical interface must be used to terminate the group of ACs on the same ES attached to remote access router.
- There can be many pseudowire subscriber logical interfaces under the same EVI, however, each pseudowire subscriber transport logical interface must be assigned a unique service instance ID.

VLAN signaled FXC on the PE Device

For a group of AC under the same EVI, the VLAN signaled FXC shares the same MPLS label on the access router, which is similar to FXC. Under VLAN signaled FXC mode, each AC is assigned a unique service instance ID and is identified by one unique normalized VLAN identifier (for single tagged frame) or VLAN identifiers (for QinQ) in the forwarding plane. Thus the VLAN signaled FXC requires that each individual point-to-point Ethernet segment to be signaled separately through its own pair of EVPN Ethernet AD per EVI routes in the control plane

Pseudowire subscriber logical interface is used to support EVPN FXC VLAN-aware service. The local service instance ID is a property associated with the pseudowire subscriber service logical interface instead of pseudowire subscriber transport logical interface. For single-active multi-homing support, you can configure each pseudowire subscriber service

logical interface with the multi-homing Ethernet segment using ESI per logical interface. There is no ESI configuration required for the pseudowire subscriber transport logical interface. Hence, the traffic is load balanced among redundant set of service edge router based on VLAN.

The following is supported on the PE device, for the interoperability with the access router which uses VLAN signaled FXC:

- There is only one pseudowire subscriber logical interface under each EVI. You must now manually configure a pair of local and remote service instance IDs on each of the pseudowire subscriber logical interface. Otherwise, the local service instance ID is auto-derived from the normalized VLAN ID(s) and the same ID is provided for remote service instance ID as well.
- A non-reserved ESI is configured for the pseudowire subscriber service logical interface, for single-active multi-homing mode.
- Each point-to-point Ethernet segment has a separate EVPN Ethernet AD per EVI route which is advertised with its Ethernet tag identifier set to the local instance ID. The Ethernet tag identifier can either be auto-derived based on the normalized VLAN ID(s) or manually configured.

VLAN Tagging

Pseudowire subscriber logical interface supports untagged, single tagged and double tagged frames. For interoperability with EVPN FXC, support for demux on single VLAN identifier or double VLANs (QinQ) for pseudowire subscriber logical interface is a must.

Signaling the Optional Bits Introduced for EVPN FXC

Both M-bit and V-bit are optional bits in the Layer 2 extended community for EVPN FXC. Currently, only the M-bit is signaled in the EVPN Layer 2 extended community, to indicate VLAN-unaware or VLAN signaled FXC.

Access Side Multi-homing

Access Side Multi-homing with Service Side Single-homing

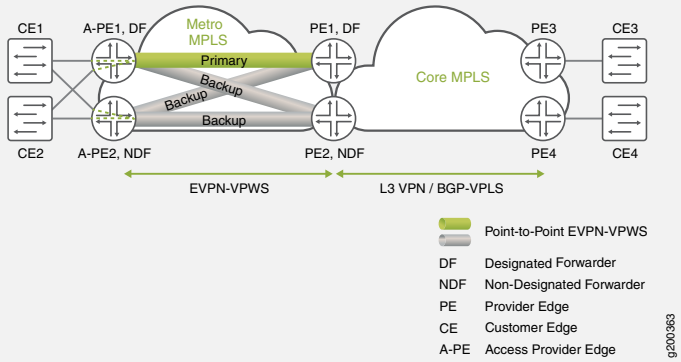
Currently, CE multi-homing to the access routers is either single-active or all-active mode (when the access device is a MX router) while the service edge router works in single-home mode is supported.

Access Side Single-active with Service Side Single-active

As shown in Figure 64 on page 673 below, this is a typical square topology consisting of two A-PE routers, A-PE1 and A-PE2, and two service edger routers, PE1 and PE2. PE1 and PE2 work in single-active mode. A-PE1 and A-PE2 also work in single-active mode. One of the service edge routers is elected as DF and one of access routers is elected as DF. There is only one active or primary pseudowire between DF access router and DF service edge router. If one of the DFs has an access link down or suffers node failure, the NDF or backup PE becomes the DF. Hence, the existing primary pseudowire goes down and a new primary pseudowire is established among the DFs. This is done as per pseudowire

subscriber service logical interface or per access link of access node. It is not done based on PE.

Figure 64: EVPN-VPWS Pseudowire Subscriber Interface Single Active

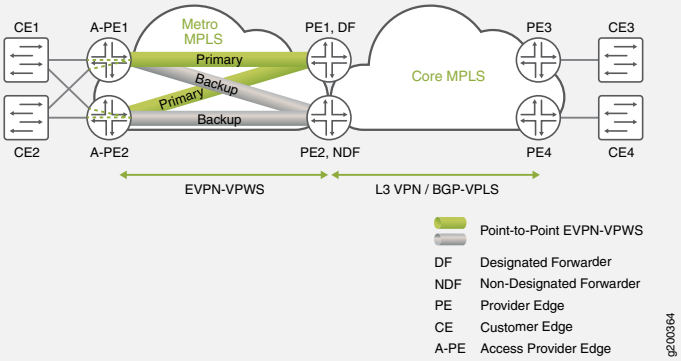


Access Side All-active with Service Side Single-active

As shown in Figure 65 on page 673 below, it is a square topology consisting of two A-PE routers, A-PE1 and A-PE2, and two service edger routers, PE1 and PE2. PE1 and PE2 work in single-active mode. A-PE1 and A-PE2 also work in all-active mode. There are two primary pseudowires from the primary or DF service router to the A-PEs. Traffic will be load balanced among A-PE1 and A-PE2.

NOTE: Always a MX router is used on the access side, for all-active mode.

Figure 65: EVPN-VPWS Pseudowire Subscriber Interface All Active



Support EVPN FXC for MX Series as Access Router

On the MX Series as access router, label per EVI is used when the service type is either EVPN FXC VLAN-unaware or EVPN FXC VLAN-aware. It is a mandatory that each AC under the same EVI must be uniquely identified through the normalized VLAN(s).

VLAN-Unaware Support on the Access Router

Under this mode of operation, all ACs are grouped separately on the basis of ESI. There is one group for all single-homed ACs and one group for each multi-homed Ethernet segment. ACs belong to the same group share the same local service instance ID. For a given EVI, the service instance ID is always unique. The service instance ID for each group must be configured manually.

VLAN-Aware Support on the Access Router

Under this mode of operation, each AC is assigned to a unique service instance ID. This service instance ID can be either manual configured or auto-derived based on the normalized VLAN identifiers.

Overview of Pseudowire Subscriber Logical Interface Support on VPWS with EVPN

Currently, Junos OS only supports pseudowire subscriber logical interface to be used with Layer 2 circuit or Layer 2 VPN for pseudowire headend termination. An Ethernet VPN (EVPN) enables you to connect dispersed customer sites using a Layer 2 virtual bridge. Virtual private wire service (VPWS) Layer 2 VPNs employ Layer 2 services over MPLS to build a topology of point-to-point connections that connect end customer sites in a VPN. EVPN-VPWS as a next generation of pseudowire technology brings the benefit of EVPN to point-to-point service by providing fast convergence upon node failure and link failure through its multi-homing feature. As a result, you can use EVPN-VPWS and pseudowire subscriber interface for headend termination into different services.

The pseudowire headend termination support with pseudowire subscriber interface works with EVPN-VPWS Flexible Cross Connect (FXC) in the vMX (Virtual MX) scale out solution. The vMX scale out support with EVPN-VPWS FXC is terminated into Layer 3 VRF or BGP-VPLS through pseudowire subscriber interface on vMX

NOTE: **The multi-home scenario includes support for either two or more than two PEs.**

- Benefits of Pseudowire Headend Termnation (PWHT) with EVPN-VPWS Pseudowires: on page 674
- Pseudowire Subscriber Logical Interface Support for EVPN-VPWS on page 675
- vMX Scale Out on page 680

Benefits of Pseudowire Headend Termnation (PWHT) with EVPN-VPWS Pseudowires:

- PWHT framework can be used to attach remote CEs (reachable via metro aggregation network) to the services (for example Layer 3 VPN) available on service PE in a similar way to CEs which are directly attached to service PEs.
- Multiple VLANs can be transported inside pseudowire and demultiplexed to different services on service PE using pseudowire subscriber logical interfaces just like VLANs on regular physical interfaces.

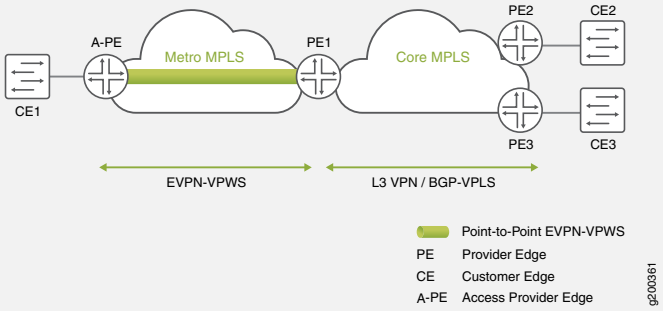
- PWHT with EVPN-VPWS as transport pseudowire brings further numerous benefits; for example unified BGP based control plane for all services in metro aggregation and core network, or redundant active-standby transport connectivity between multiple service PEs and multiple aggregation nodes.

Pseudowire Subscriber Logical Interface Support for EVPN-VPWS

Figure 66 on page 675 illustrates an EVPN-VPWS network with pseudowire service. A-PE (Access Provider Edge) is the access router and PE1 is the service edge router. A pseudowire subscriber logical interface is used on PE1 to terminate the pseudowire established by EVPN-VPWS into either Layer 3 VPN or Layer 2 BGP-VPLS service. There is only one single service edge router to terminate the pseudowire, in this scenario.

For a given pseudowire subscriber logical interface, for example ps0, only the transport interface of the pseudowire subscriber logical interface, that is ps0.0, is used as an access interface for the EVPN-VPWS on PE1. The service interfaces of ps0, that is ps0.1 to ps0.n, are used at the service side (either under Layer 3 VRF or BGP-VPLS instance).

Figure 66: EVPN-VPWS with Pseudowire Subscriber Interface



NOTE: Prior to Junos OS 18.2 Release, the only encapsulation type that pseudowire subscriber transport logical interface (as an access interface for the pseudowire) supports for Layer 2 circuit and Layer 2 VPN is ethernet-ccc. This encapsulation type will remain the same for EVPN-VPWS with pseudowire subscriber logical interface support as well.

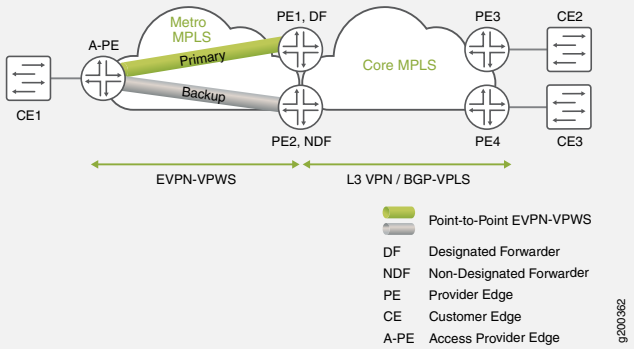
Single Pseudowire Headend Termination

The pseudowire subscriber transport logical interface is used as the access interface for EVPN-VPWS instance. EVPN establishes a point-to-point Ethernet service with pseudowire subscriber transport logical interface as an access interface in the control plane.

Active-Standby Pseudowire Headend Termination

As shown in Figure 67 on page 676, to achieve resilience for pseudowire headend termination into Layer 3 VPN or BGP-VPLS, you can use a pair of redundant pseudowires on the EVPN-VPWS side. Also, in Figure 67 on page 676, Layer 3 VPN or BGP-VPLS is treated as if it is multi-homed to the set of redundant service edge routers, PE1 and PE2. PE1 and PE2 work in active-standby mode for EVPN-VPWS. One of them is elected as the primary PE per EVPN normal designated forwarder (DF) election procedure. Only the primary PE is used to forward Layer 2 traffic.

Figure 67: EVPN-VPWS Active/standby with Pseudowire Subscriber Interface



Following are the reasons for the pseudowire failure, when redundancy is provided by the EVPN-VPWS active-standby interface:

- Service edge router node failure.
- Pseudowire subscriber transport logical interface failure on the primary PE.
- Failure on the path towards the primary PE.

NOTE: If any of the above failure cases are detected, the backup PE takes over to become the primary PE. As a result, the customer traffic from the access router, A-PE, is switched to the new primary PE.

The EVPN-VPWS also supports the service side of the network multi-homing to EVPN-VPWS through the pseudowire subscriber logical interface in an active-standby mode.

To achieve active-standby pseudowire headend termination, the following is required:

- *Ethernet segment identifier (ESI)* support on the pseudowire subscriber transport logical interface.
- Synchronizing data path between active pseudowire and Layer 3 VPN.

- MAC flush in the BGP-VPLS when the active-standby pseudowires switch. MAC flush is triggered when the active service edge node suffers a node failure.

ES Support on Pseudowire Subscriber Transport Logical Interface

An Ethernet segment must have a unique nonzero identifier, called the *Ethernet segment identifier (ESI)*. The ESI is encoded as a 10-octet integer. When manually configuring an ESI value, the most significant octet, known as the *type byte*, must be 00. ESI is assigned to pseudowire subscriber transport logical interfaces associated with the PE1 and PE2 the same way as in EVPN multi-homing.

To configure the active-standby mode for the pseudowire subscriber transport logical interface, include the ESI value and the **single-active** statement under the [**edit interfaces**] hierarchy level.

DF Election and Pseudowire Subscriber Transport Logical Interface

Designated forwarder(DF)—This is the designated forwarder for forwarding the current traffic. The DF election procedure ensures each VLAN is associated with single PE acting in DF role.

Synchronizing Data Path between EVPN-VPWS and Layer 3 VPN or BGP-VPLS

There is a need for the data path coordination between the point-to-point pseudowire and the services side of the network (Layer 3 VPN or BGP-VPLS), with active-standby pseudowire headend termination. This is to select the same service edge router for passing the current traffic between the point-to-point pseudowire and Layer 3 VPN or BGP-VPLS domain.

EVPN-VPWS determines the primary path, for both Layer 3 VPN and BGP-VPLS, based on its DF election result. Both the primary and backup PEs for EVPN-VPWS then influence the services side of network to use the primary PE for data traffic.

Layer 3 VPN

When pseudowires are headend terminated into the Layer 3 VPN in an active-standby mode, the primary and the backup PEs use routing-policy and policy-condition to increase or decrease the Local Preference (LP). As a result, the BGP path selection algorithm is carried out on the Layer 3 side and picks up the primary PE.

BGP-VPLS

EVPN-VPWS establishes active-standby pseudowires based on its DF election. The active-standby pseudowires are essentially the two redundant spoke pseudowires attached to BGP-VPLS instance. The BGP-VPLS PEs remain single homed PEs and data plane learning happens through the active pseudowire in the EVPN-VPWS primary path.

NOTE:

- If the pseudowire subscriber service logical interface that belongs to a BGP-VPLS instance is in down state, this does not trigger the active and standby pseudowire switch on the EVPN-VPWS side. This is because the 1 to many relationships between the point-to-point pseudowire and the BGP-VPLS. This may cause traffic loss for traffic coming from the VPLS to the point-to-point pseudowire direction.
- BGP-VPLS does not trigger MAC flush when its access link to the CE device goes up or down. Further, since multi-homed EVPN-VPWS pseudowire headend termination into BGP-VPLS is based on single-homed mode on BGP-VPLS side, MAC flush (achieved via manipulation of F-bit in BGP-VPLS messages) associated with multi-homed BGP-VPLS mode does not apply here, neither. Therefore, when there is DF or NDF state change on EVPN-VPWS side, or if the pseudowire subscriber service logical interface that belongs to a BGP-VPLS instance goes up or down, MAC flush is not triggered. Traffic black holing might occur until expiration of MAC aging timer, unless there is constant traffic from the CE behind the EVPN-VPWS to BGP-VPLS. MAC flush is triggered only during PE node failures.

The following configuration example shows how condition manager is used for Layer 3 VPN routing instance through vrf-export when active-standby EVPN-VPWS terminates into Layer 3 VPN service.

```
[edit]
routing-instances {
   l3vpn_1 {
      instance-type vrf;
      interface ps0.1;
      interface lo0.1;
      route-distinguisher 2.2.2.3:6500;
      vrf-import l3vpn1_import;
      vrf-export l3vpn1_export;
      vrf-table-label;
      protocols {
         group toPW-CE1 {
            type external;
            export send-direct;
            peer-as 1;
            neighbor 10.1.1.1;
         }
      }
   }
}
policy-options {
   policy-statement l3vpn_1_export {
      term 1 {
         from condition primary;
         then {
            local-preference add 300;
            community set l3vpn_1;
```

```
            accept;
        }
    }
    term 2 {
        from condition standby;
        then {
            community set l3vpn_1;
            accept;
        }
    }
}
policy-statement l3vpn1_import {
    term 1 {
        from community l3vpn_1;
        then accept;
    }
    term default {
        then reject;
    }
}
community l3vpn_1 members target:65056:100;
condition primary {
    if-route-exists {
        address-family {
            ccc {
                ps0.0;
                table mpls.0;
            }
        }
    }
}
condition standby {
    if-route-exists {
        address-family {
            ccc {
                ps0.0;
                table mpls.0;
                standby;
            }
        }
    }
}
}
```

Following is the BGP-VPLS instance configuration when EVPN-VPWS terminates into the BGP-VPLS:

```
[edit]
routing-instances {
    vpls-1 {
        instance-type vpls;
        interface ps0.1;
        route-distinguisher 100:2;
        vrf-target target:100:100;
        protocols {
```

```
        vpls {
          site ce3 {
            site-identifier 3;
          }
        }
      }
    }
  }
```

Active-Active Pseudowire Headend Termination

Active-active pseudowire headend termination is supported only in Layer 3 VPN and not supported in BGP-VPLS.

vMX Scale Out

The vMX router is a virtual version of the MX Series 5G Universal Routing Platform. Like the MX Series router, the vMX router runs the Junos operating system (Junos OS) and supports Junos OS packet handling and forwarding modeled after the Trio chipset. The vMX instance contains two separate virtual machines (VMs), one for the virtual forwarding plane (VFP) and one for the virtual control plane (VCP). The VFP VM runs the virtual Trio forwarding plane software and the VCP VM runs Junos OS.

The vMX can now be used to scale out the bandwidth, achieve service isolation, and resilience. As the service edge router, vMX supports pseudowire headend termination with active and standby mode.

vMX

A vMX can have active and backup VCP with one or more VFPs. The VCP is the RE and the VFP is the line-card. The communication between VCP and VFP is facilitated through vRouter, if the VCP and VFP reside on the same server. Else, it is through the external network to which the servers are connected. There is no direction communication between VFPs.

Service Isolation

For a given point-to-point pseudowire with only one VFP per vMX, the data traffic is handled by one VFP for both ingress and egress traffic.

Active-Standby Pseudowire Headend Termination into VMX

The vMX acts as a regular MX for the overlay point-to-point Ethernet service. Similar to physical MX router, redundancy is achieved by using multiple vMXs, and only active-standby pseudowire headend termination is supported. The pseudowire is terminated at the loopback IP address that is used as the protocol nexthop address for the pseudowire.

Related Documentation

- Example: Configuring VPWS with EVPN Signaling Mechanisms on page 683

Configuring VPWS with EVPN Signaling Mechanisms

The virtual private wire service (VPWS) with Ethernet VPN (EVPN) provides single-active or all-active multihoming capabilites along with support for Inter-AS options associated with BGP-signaled VPNs. The VPWS service identifiers identify the endpoints of the EVPN-VPWS network. Each provider edge (PE) router in EVPN-VPWS network is configured with local and remote VPWS service identifiers.

Before you configure VPWS service with EVPN mechanisms, you must do the following:

1. Configure the router interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Configure OSPF.
4. Configure a BGP internal group.
5. Configure ISIS.
6. Include the EVPN signaling network layer reachability information (NLRI) to the internal BGP group.
7. Configure LDP.
8. Configure MPLS.
9. Configure MPLS LSP or GRE tunnels.
10. Configure EVPN all-active multihoming and EVPN single-active multihoming.

To configure a PE device with a VPWS service identifier in an EVPN-VPWS network, do the following:

1. Configure the EVPN-VPWS routing instance.

   ```
   [edit routing-instances]
   user@PE# set evpn-vpws-instance instance-type instance-type-value
   ```

   For example, configure routing instance vpws1004 with instance type evpn-vpws.

   ```
   [edit routing-instances]
   user@PE# set vpws1004 instance-type evpn-vpws
   ```

2. Configure the interface names for the EVPN-VPWS routing instance.

   ```
   [edit routing-instances]
   user@PE# set evpn-vpws-instance interface interface-name
   ```

   For example, configure interface ge-0/0/1.1004 for the EVPN-VPWS instance vpws1004.

   ```
   [edit routing-instances]
   user@PE# set vpws1004 interface ge-0/0/1.1004
   ```

3. Configure the route distinguisher for the EVPN-VPWS routing instance.

```
[edit routing-instances]
user@PE# set evpn-vpws-instance route-distinguisher route-distinguisher-value
```

   For example, configure route distinguisher 10.255.0.1:100 for EVPN-VPWS routing instance vpws1004.

```
[edit routing-instances]
user@PE# set vpws1004 route-distinguisher 10.255.0.1:100
```

4. Configure the VPN routing and forwarding (VRF) target community for the EVPN-VPWS routing instance.

```
[edit routing-instances]
user@PE# set evpn-vpws-instance vrf-target vrf-target
```

   For example, configure VRF target target:100:1004 for EVPN-VPWS routing instance vpws1004.

```
[edit routing-instances]
user@PE# set vpws1004 vrf-target target:100:1004
```

5. Configure the interface of a routing instance with local and remote service identifiers. These identifiers identify the PE routers that forward and receive the traffic in the EVPN-VPWS network. The local service identifier is used to identify the PE router that is forwarding the traffic, and the remote service identifier is used to identify the PE router that is receiving the traffic in the network.

```
[edit routing-instances evpn-vpws-instance protocols evpn interface interface-name]
user@PE# set vpws-service-id local local-service-id
user@PE# set vpws-service-id remote remote-service-id
```

   For example, configure the interface ge-0/0/1.1004 with the local and remote service identifiers 1004 and 2004 for EVPN-VPWS routing instance vpws1004.

```
[edit routing-instances vpws1004 protocols evpn interface ge-0/0/1.1004]
user@PE# set vpws-service-id local 1004
user@PE# set vpws-service-id remote 2004
```

Related Documentation

- Overview of VPWS with EVPN Signaling Mechanisms on page 666

Example: Configuring VPWS with EVPN Signaling Mechanisms

This example shows how to implement Virtual Private Wire Service (VPWS) with Ethernet Virtual Private Network (EVPN) signaling. The use of EVPN signaling provides single-active or all-active multihoming capabilities for BGP-signaled VPNs.

Requirements

This example uses the following hardware and software components:

- Four MX Series routers acting as provider edge (PE) devices, running Junos OS Release 17.1 or later
- Two customer edge (CE) devices (MX Series routers are used in this example)

Overview and Topology

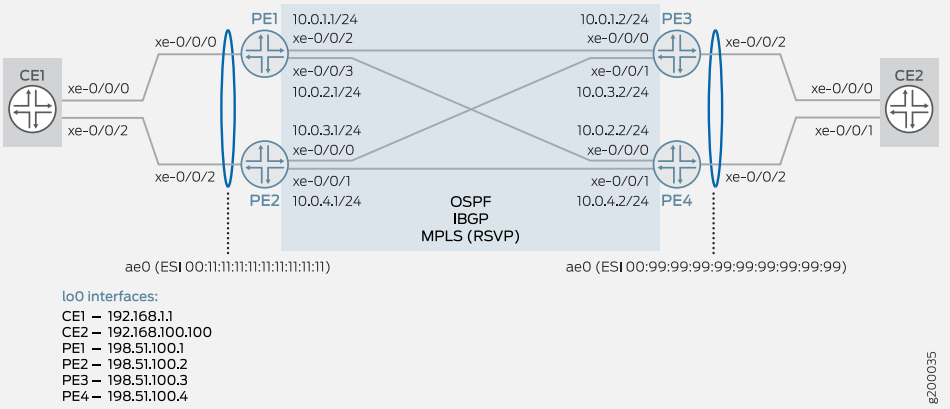
VPWS employs Layer 2 VPN services over MPLS to build a topology of point-to-point connections that connect end customer sites. EVPN enables you to connect dispersed customer sites using a Layer 2 virtual bridge. Starting with Junos OS Release 17.1, these two elements can be combined to provide an EVPN-signaled VPWS.

The **vpws-service-id** statement identifies the endpoints of the EVPN-VPWS based on **local** and **remote** service identifiers configured on the PE routers in the network. These endpoints are autodiscovered using BGP-based EVPN signaling to exchange the service identifier labels.

Topology

This example uses the topology shown in Figure 68 on page 684, consisting of four PE routers and two CE routers. Router CE1 is multihomed to Routers PE1 and PE2; Router CE2 is multhomed to Routers PE3 and PE4.

Figure 68: VPWS with EVPN Signaling



The following configuration elements are used in this scenario:

- CE devices:
  - Aggregated Ethernet (AE) interface towards related PE devices
- PE devices:
  - AE interface, with EVPN segment identifier (ESI), towards related CE device
  - OSPF and IBGP in the core
  - MPLS LSPs using RSVP in the core
  - Per-packet load balancing
  - Routing instance using instance type **evpn-vpws**, and the **vpws-service-id** statement to define the local and remote endpoints.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level.

CE1

```
set interfaces xe-0/0/0 gigether-options 802.3ad ae0
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to PE1/2"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 100 encapsulation vlan-bridge
set interfaces ae0 unit 100 vlan-id 1000
set interfaces lo0 unit 0 family inet address 192.168.1.1/32
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
```

```
set bridge-domains bd100 domain-type bridge
set bridge-domains bd100 vlan-id 1000
set bridge-domains bd100 interface ae0.100
```

CE2

```
set interfaces xe-0/0/0 gigether-options 802.3ad ae0
set interfaces xe-0/0/1 gigether-options 802.3ad ae0
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to PE3/4"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 unit 100 encapsulation vlan-bridge
set interfaces ae0 unit 100 vlan-id 1000
set interfaces lo0 unit 0 family inet address 192.168.100.100/32
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set bridge-domains bd100 domain-type bridge
set bridge-domains bd100 vlan-id 1000
set bridge-domains bd100 interface ae0.100
```

PE1

```
set interfaces xe-0/0/0 description "to CE1"
set interfaces xe-0/0/0 gigether-options 802.3ad ae0
set interfaces xe-0/0/2 unit 0 description "to PE3"
set interfaces xe-0/0/2 unit 0 family inet address 10.0.1.1/24
set interfaces xe-0/0/2 unit 0 family mpls
set interfaces xe-0/0/3 unit 0 description "to PE4"
set interfaces xe-0/0/3 unit 0 family inet address 10.0.2.1/24
set interfaces xe-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.1/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE1"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc
set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/2.0
set protocols ospf area 0.0.0.0 interface xe-0/0/3.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.1
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.2
set protocols bgp group IBGP neighbor 198.51.100.3
set protocols bgp group IBGP neighbor 198.51.100.4
set protocols rsvp interface xe-0/0/2.0
set protocols rsvp interface xe-0/0/3.0
set protocols mpls interface xe-0/0/2.0
```

```
set protocols mpls interface xe-0/0/3.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE1toPE3 to 198.51.100.3
set protocols mpls label-switched-path PE1toPE4 to 198.51.100.4
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.1:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  local 1111
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  remote 9999
```

PE2

```
set interfaces xe-0/0/0 unit 0 description "to PE3"
set interfaces xe-0/0/0 unit 0 family inet address 10.0.3.1/24
set interfaces xe-0/0/0 unit 0 family mpls
set interfaces xe-0/0/1 unit 0 description "to PE4"
set interfaces xe-0/0/1 unit 0 family inet address 10.0.4.1/24
set interfaces xe-0/0/1 unit 0 family mpls
set interfaces xe-0/0/2 description "to CE1"
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set interfaces lo0 unit 0 family inet address 198.51.100.2/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE1"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc
set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.2
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.1
set protocols bgp group IBGP neighbor 198.51.100.3
set protocols bgp group IBGP neighbor 198.51.100.4
set protocols rsvp interface xe-0/0/0.0
set protocols rsvp interface xe-0/0/1.0
set protocols mpls interface xe-0/0/0.0
set protocols mpls interface xe-0/0/1.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE2toPE3 to 198.51.100.3
set protocols mpls label-switched-path PE2toPE4 to 198.51.100.4
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
```

```
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.2:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  local 1111
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  remote 9999
```

PE3

```
set interfaces xe-0/0/0 unit 0 description "to PE1"
set interfaces xe-0/0/0 unit 0 family inet address 10.0.1.2/24
set interfaces xe-0/0/0 unit 0 family mpls
set interfaces xe-0/0/1 unit 0 description "to PE2"
set interfaces xe-0/0/1 unit 0 family inet address 10.0.3.2/24
set interfaces xe-0/0/1 unit 0 family mpls
set interfaces xe-0/0/2 description "to CE1"
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set interfaces lo0 unit 0 family inet address 198.51.100.3/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE2"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:99:99:99:99:99:99:99:99:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc
set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.3
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.1
set protocols bgp group IBGP neighbor 198.51.100.2
set protocols bgp group IBGP neighbor 198.51.100.4
set protocols rsvp interface xe-0/0/0.0
set protocols rsvp interface xe-0/0/1.0
set protocols mpls interface xe-0/0/0.0
set protocols mpls interface xe-0/0/1.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE3toPE1 to 198.51.100.1
set protocols mpls label-switched-path PE3toPE2 to 198.51.100.2
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.3:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  local 9999
```

```
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  remote 1111
```

PE4

```
set interfaces xe-0/0/0 unit 0 description "to PE1"
set interfaces xe-0/0/0 unit 0 family inet address 10.0.2.2/24
set interfaces xe-0/0/0 unit 0 family mpls
set interfaces xe-0/0/1 unit 0 description "to PE2"
set interfaces xe-0/0/1 unit 0 family inet address 10.0.4.2/24
set interfaces xe-0/0/1 unit 0 family mpls
set interfaces xe-0/0/2 description "to CE1"
set interfaces xe-0/0/2 gigether-options 802.3ad ae0
set interfaces lo0 unit 0 family inet address 198.51.100.4/32
set chassis aggregated-devices ethernet device-count 1
set interfaces ae0 description "to CE2"
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:99:99:99:99:99:99:99:99:99
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
set interfaces ae0 unit 100 encapsulation vlan-ccc
set interfaces ae0 unit 100 vlan-id 1000
set protocols ospf area 0.0.0.0 interface xe-0/0/0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set routing-options autonomous-system 65000
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.4
set protocols bgp group IBGP family evpn signaling
set protocols bgp group IBGP neighbor 198.51.100.1
set protocols bgp group IBGP neighbor 198.51.100.2
set protocols bgp group IBGP neighbor 198.51.100.3
set protocols rsvp interface xe-0/0/0.0
set protocols rsvp interface xe-0/0/1.0
set protocols mpls interface xe-0/0/0.0
set protocols mpls interface xe-0/0/1.0
set protocols mpls no-cspf
set protocols mpls label-switched-path PE4toPE1 to 198.51.100.1
set protocols mpls label-switched-path PE4toPE2 to 198.51.100.2
set policy-options policy-statement LB then load-balance per-packet
set routing-options forwarding-table export LB
set routing-instances EVPN-VPWS instance-type evpn-vpws
set routing-instances EVPN-VPWS interface ae0.100
set routing-instances EVPN-VPWS route-distinguisher 198.51.100.4:11
set routing-instances EVPN-VPWS vrf-target target:100:11
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  local 9999
set routing-instances EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  remote 1111
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

NOTE: Only Router PE1 is shown here. Repeat this procedure for all other PE devices, using the appropriate interface names, addresses, and other parameters for each device.

The step-by-step procedure for CE devices is not shown.

To configure Router PE1:

1. Configure the CE-facing interface to be part of the ae0 bundle.

   The second interface for the AE bundle will be configured on the other local PE device.

   ```
   [edit interfaces]
   user@PE1# set xe-0/0/0 description "to CE1"
   user@PE1# set xe-0/0/0 gigether-options 802.3ad ae0
   ```

2. Configure the core-facing interfaces toward Routers PE3 and PE4.

   Be sure to include the MPLS protocol family.

   ```
   [edit interfaces]
   user@PE1# set xe-0/0/2 unit 0 description "to PE3"
   user@PE1# set xe-0/0/2 unit 0 family inet address 10.0.1.1/24
   user@PE1# set xe-0/0/2 unit 0 family mpls
   user@PE1# set xe-0/0/3 unit 0 description "to PE4"
   user@PE1# set xe-0/0/3 unit 0 family inet address 10.0.2.1/24
   user@PE1# set xe-0/0/3 unit 0 family mpls
   ```

3. Configure the loopback interface.

   ```
   [edit interfaces]
   user@PE1# set lo0 unit 0 family inet address 198.51.100.1/32
   ```

4. Define the number of aggregated Ethernet interfaces to be supported on the device.

   ```
   [edit chassis]
   user@PE1# set aggregated-devices ethernet device-count 1
   ```

5. Configure the ae0 interface.

   Alternate VLAN tagging and encapsulation options can be used, depending on your needs.

```
[edit interfaces]
user@PE1# set ae0 description "to CE1"
user@PE1# set ae0 flexible-vlan-tagging
user@PE1# set ae0 encapsulation flexible-ethernet-services
user@PE1# set ae0 unit 100 encapsulation vlan-ccc
user@PE1# set ae0 unit 100 vlan-id 1000
```

6. Assign an Ethernet segment identifier (ESI) value to the ae0 interface and enable EVPN active-active multihoming.

```
[edit interfaces]
user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11:11
user@PE1# set ae0 esi all-active
```

7. Configure link aggregation control protocol (LACP) for the ae0 interface.

   The system ID used here must be the same on both local PE devices.

```
[edit interfaces]
user@PE1# set ae0 aggregated-ether-options lacp active
user@PE1# set ae0 aggregated-ether-options lacp system-id 00:00:00:00:00:01
```

8. Enable OSPF on the core-facing (and loopback) interfaces.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/2.0
user@PE1# set ospf area 0.0.0.0 interface xe-0/0/3.0
user@PE1# set ospf area 0.0.0.0 interface lo0.0
```

9. Configure an IBGP mesh with the other PE devices, using EVPN for signaling.

```
[edit routing-options]
user@PE1# set autonomous-system 65000
[edit protocols]
user@PE1# set bgp group IBGP type internal
user@PE1# set bgp group IBGP local-address 198.51.100.1
user@PE1# set bgp group IBGP family evpn signaling
user@PE1# set bgp group IBGP neighbor 198.51.100.2
user@PE1# set bgp group IBGP neighbor 198.51.100.3
user@PE1# set bgp group IBGP neighbor 198.51.100.4
```

10. Enable RSVP on the core-facing interfaces.

```
[edit protocols]
user@PE1# set rsvp interface xe-0/0/2.0
user@PE1# set rsvp interface xe-0/0/3.0
```

11. Enable MPLS on the core-facing interfaces, and configure LSPs to the remote PE devices.

    For this example, be sure to disable CSPF.

```
[edit protocols]
user@PE1# set mpls interface xe-0/0/2.0
user@PE1# set mpls interface xe-0/0/3.0
user@PE1# set mpls no-cspf
user@PE1# set mpls label-switched-path PE1toPE3 to 198.51.100.3
user@PE1# set mpls label-switched-path PE1toPE4 to 198.51.100.4
```

12. Configure load balancing.

```
[edit policy-options]
user@PE1# set policy-statement LB then load-balance per-packet
[edit routing-options]
user@PE1# set forwarding-table export LB
```

13. Configure a routing instance using the **evpn-vpws** instance type. Add the AE (CE-facing) interface configured earlier, as well as a route distinguisher and VRF target.

    In EVPN terms, this is an EVPN instance (EVI).

```
[edit routing-instances]
user@PE1# set EVPN-VPWS instance-type evpn-vpws
user@PE1# set EVPN-VPWS interface ae0.100
user@PE1# set EVPN-VPWS route-distinguisher 198.51.100.1:11
user@PE1# set EVPN-VPWS vrf-target target:100:11
```

14. In the routing instance, enable EVPN and add the AE interface. Then associate local and remote VPWS identifiers to the interface.

```
[edit routing-instances]
user@PE1# set EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id local
  1111
user@PE1# set EVPN-VPWS protocols evpn interface ae0.100 vpws-service-id
  remote 9999
```

Results

From configuration mode, confirm your configuration. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit ]
user@PE1# show chassis
aggregated-devices {
  ethernet {
```

```
    device-count 1;
  }
}
```

```
[edit ]
user@PE1# show interfaces
xe-0/0/0 {
  description "to CE1";
  gigether-options {
    802.3ad ae0;
  }
}
xe-0/0/2 {
  unit 0 {
    description "to PE3";
    family inet {
      address 10.0.1.1/24;
    }
    family mpls;
  }
}
xe-0/0/3 {
  unit 0 {
    description "to PE4";
    family inet {
      address 10.0.2.1/24;
    }
    family mpls;
  }
}
ae0 {
  description "to CE1";
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  esi {
    00:11:11:11:11:11:11:11:11:11;
    all-active;
  }
  aggregated-ether-options {
    lacp {
      active;
      system-id 00:00:00:00:00:01;
    }
  }
  unit 100 {
    encapsulation vlan-ccc;
    vlan-id 1000;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 198.51.100.1/32;
    }
```

```
    }
}
```

```
[edit ]
user@PE1# show routing-options
autonomous-system 65000;
forwarding-table {
    export LB;
}
```

```
user@PE1# show protocols
rsvp {
    interface xe-0/0/2.0;
    interface xe-0/0/3.0;
}
mpls {
    no-cspf;
    label-switched-path PE1toPE3 {
        to 198.51.100.3;
    }
    label-switched-path PE1toPE4 {
        to 198.51.100.4;
    }
    interface xe-0/0/2.0;
    interface xe-0/0/3.0;
}
bgp {
    group IBGP {
        type internal;
        local-address 198.51.100.1;
        family evpn {
            signaling;
        }
        neighbor 198.51.100.2;
        neighbor 198.51.100.3;
        neighbor 198.51.100.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface xe-0/0/2.0;
        interface xe-0/0/3.0;
        interface lo0.0;
    }
}
```

```
[edit ]
user@PE1# show policy-options
policy-statement LB {
    then {
        load-balance per-packet;
    }
}
```

```
[edit ]
user@PE1# show routing-instances
EVPN-VPWS {
   instance-type evpn-vpws;
   interface ae0.100;
   route-distinguisher 198.51.100.1:11;
   vrf-target target:100:11;
   protocols {
      evpn {
         interface ae0.100 {
            vpws-service-id {
               local 1111;
               remote 9999;
            }
         }
      }
   }
}
```

If you are done configuring the device, enter **commit** from the configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying Agregated Ethernet Interfaces and LACP

Purpose Verify that the AE interfaces are up and properly.

Action Verify that the AE interfaces are up, and LACP connections are established between the PE devices and their related CE device..

```
user@CE1> show lacp interfaces extensive
```

```
Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      xe-0/0/0       Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      xe-0/0/0     Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
      xe-0/0/2       Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      xe-0/0/2     Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:        Receive State  Transmit State          Mux State
      xe-0/0/0                  Current   Fast periodic Collecting distributing
```

```
      xe-0/0/2                  Current   Fast periodic Collecting distributing
    LACP info:        Role     System            System       Port     Port
Port
                             priority         identifier   priority   number
 key
      xe-0/0/0       Actor        127  44:f4:77:99:e3:c0        127        1
   1
      xe-0/0/0     Partner        127  00:00:00:00:00:01        127        1
   1
      xe-0/0/2       Actor        127  44:f4:77:99:e3:c0        127        2
   1
      xe-0/0/2     Partner        127  00:00:00:00:00:01        127        1
   1
```

user@PE1> **show lacp interfaces extensive**

```
Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      xe-0/0/0       Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      xe-0/0/0     Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:        Receive State  Transmit State          Mux State
      xe-0/0/0                  Current   Fast periodic Collecting distributing
    LACP info:        Role     System            System       Port     Port
Port
                             priority         identifier   priority   number
 key
      xe-0/0/0       Actor        127  00:00:00:00:00:01        127        1
   1
      xe-0/0/0     Partner        127  44:f4:77:99:e3:c0        127        1
   1
```

user@PE2> **show lacp interfaces extensive**

```
Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      xe-0/0/2       Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      xe-0/0/2     Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:        Receive State  Transmit State          Mux State
      xe-0/0/2                  Current   Fast periodic Collecting distributing
    LACP info:        Role     System            System       Port     Port
Port
                             priority         identifier   priority   number
 key
      xe-0/0/2       Actor        127  00:00:00:00:00:01        127        1
   1
      xe-0/0/2     Partner        127  44:f4:77:99:e3:c0        127        2
   1
```

Meaning The AE interface on each device is up, and there are active LACP connections between the CE device and its local PE devices. Note also that the system ID configured on the PE devices, **00:00:00:00:00:01** (and shown on the PE device outputs as the **Actor**), matches the **Partner** system ID value on the CE device.

Verifying OSPF

Purpose Verify that OSPF is working properly.

Action Verify that OSPF has adjacencies established with its remote neighbors.

```
user@PE1> show ospf neighbor

Address          Interface              State     ID               Pri  Dead
10.0.1.2  #PE3#  xe-0/0/2.0             Full      198.51.100.3     128    37
10.0.2.2  #PE4#  xe-0/0/3.0             Full      198.51.100.4     128    33
```

```
user@PE3> show ospf neighbor

Address          Interface              State     ID               Pri  Dead
10.0.1.1  #PE1#  xe-0/0/0.0             Full      198.51.100.1     128    34
10.0.3.1  #PE2#  xe-0/0/1.0             Full      198.51.100.2     128    34
```

Meaning Adjacencies have been established with remote neighbors.

Verifying BGP

Purpose Verify that BGP is working properly.

Action Verify that IBGP has peerings established with its neighbors using EVPN signaling.

```
user@PE1> show bgp summary

Groups: 1 Peers: 3 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                       7          4          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
198.51.100.2  #PE2#   65000         12          5       0       0        3:03
Establ
  bgp.evpn.0: 0/3/3/0
  EVPN-VPWS.evpn.0: 0/2/2/0
  __default_evpn__.evpn.0: 0/1/1/0
198.51.100.3  #PE3#   65000         11          9       0       0        3:03
Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
198.51.100.4  #PE4#   65000          9          4       0       0        1:56
Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
```

```
user@PE3> show bgp summary

Groups: 1 Peers: 3 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                       7          4          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
198.51.100.1  #PE1#   65000         17         11       0       0        5:09
Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
198.51.100.2  #PE2#   65000         17         14       0       0        5:04
Establ
  bgp.evpn.0: 2/2/2/0
  EVPN-VPWS.evpn.0: 2/2/2/0
  __default_evpn__.evpn.0: 0/0/0/0
198.51.100.4  #PE34   65000         13          8       0       0        4:02
Establ
  bgp.evpn.0: 0/3/3/0
  EVPN-VPWS.evpn.0: 0/2/2/0
  __default_evpn__.evpn.0: 0/1/1/0
```

Meaning EVPN-signaled IBGP peerings have been established with all neighbors.

Verifying MPLS

Purpose Verify that MPLS is working properly.

Action Verify that MPLS LSPs are established with remote neighbors.

```
user@PE1> show mpls lsp

Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
198.51.100.3    198.51.100.1    Up     0 *                      PE1toPE3
198.51.100.4    198.51.100.1    Up     0 *                      PE1toPE4
Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
198.51.100.1    198.51.100.4    Up       0  1 FF       3        - PE4toPE1
198.51.100.1    198.51.100.3    Up       0  1 FF       3        - PE3toPE1
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

```
user@PE3> show mpls lsp

Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
```

```
198.51.100.1    198.51.100.3    Up     0 *                      PE3toPE1
198.51.100.2    198.51.100.3    Up     0 *                      PE3toPE2
Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
198.51.100.3    198.51.100.2    Up       0  1 FF       3        - PE2toPE3
198.51.100.3    198.51.100.1    Up       0  1 FF       3        - PE1toPE3
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

Meaning LSPs have been established with remote neighbors.

Verifying the VPWS

Purpose Verify that the VPWS is established.

Action Verify that the PE devices have exchanged and learned service identifiers, and established the VPWS.

```
user@PE1> show evpn vpws-instance
```

```
Instance: EVPN-VPWS
  Route Distinguisher: 198.51.100.1:11
  Number of local interfaces: 1 (1 up)

    Interface name  ESI                             Mode            Role
Status
    ae0.100         00:11:11:11:11:11:11:11:11:11  all-active      Primary    Up

        Local SID: 1111 Advertised Label: 300496
        Remote SID: 9999
            PE addr         ESI                             Label  Mode
Role     TS                      Status
            198.51.100.3    00:99:99:99:99:99:99:99:99:99 300656 all-active
Primary  2017-05-12 07:30:01.863 Resolved
            198.51.100.4    00:99:99:99:99:99:99:99:99:99 300704 all-active
Primary  2017-05-12 07:31:23.804 Resolved

    Fast Convergence Information
    ESI: 00:99:99:99:99:99:99:99:99:99 Number of PE nodes: 2
        PE: 198.51.100.3     #PE3#
            Advertised SID: 9999
        PE: 198.51.100.4     #PE4#
            Advertised SID: 9999
```

```
user@PE3> show evpn vpws-instance
```

```
Instance: EVPN-VPWS
  Route Distinguisher: 198.51.100.3:11
  Number of local interfaces: 1 (1 up)

    Interface name  ESI                             Mode            Role
Status
    ae0.100         00:99:99:99:99:99:99:99:99:99  all-active      Primary    Up

        Local SID: 9999 Advertised Label: 300656
        Remote SID: 1111
            PE addr         ESI                             Label  Mode
Role     TS                     Status
            198.51.100.1    00:11:11:11:11:11:11:11:11:11 300496 all-active
Primary  2017-05-12 07:30:25.702 Resolved
            198.51.100.2    00:11:11:11:11:11:11:11:11:11 300560 all-active
Primary  2017-05-12 07:30:25.711 Resolved

    Fast Convergence Information
    ESI: 00:11:11:11:11:11:11:11:11:11 Number of PE nodes: 2
        PE: 198.51.100.1     #PE1#
            Advertised SID: 1111
        PE: 198.51.100.2     #PE2#
            Advertised SID: 1111
```

Meaning The PE devices on each side of the network have advertised their service identifiers, and received the identifiers from their remote neighbors. The VPWS is established.

Verifying Route Exchange and ESI Autodiscovery

Purpose Verify that EVPN signaling is working properly.

Action Verify that autodiscovery information is being shared across the VPWS.

```
user@PE1> show route table bgp.evpn.0


bgp.evpn.0: 7 destinations, 7 routes (4 active, 0 holddown, 3 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.3:0::999999999999999999::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:03:17, localpref 100, from 198.51.100.3
                      AS path: I, validation-state: unverified
                    > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.3:11::999999999999999999::9999/304 AD/EVI
                   *[BGP/170] 00:03:18, localpref 100, from 198.51.100.3
                      AS path: I, validation-state: unverified
                    > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.4:0::999999999999999999::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:01:56, localpref 100, from 198.51.100.4
                      AS path: I, validation-state: unverified
                    > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4
1:198.51.100.4:11::999999999999999999::9999/304 AD/EVI
                   *[BGP/170] 00:01:56, localpref 100, from 198.51.100.4
```

```
                      AS path: I, validation-state: unverified
                    > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4
```

```
user@PE3> show route table bgp.evpn.0

bgp.evpn.0: 7 destinations, 7 routes (4 active, 0 holddown, 3 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::111111111111111111::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:04:53, localpref 100, from 198.51.100.1
                      AS path: I, validation-state: unverified
                    > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.1:11::11111111111111111::1111/304 AD/EVI
                   *[BGP/170] 00:04:53, localpref 100, from 198.51.100.1
                      AS path: I, validation-state: unverified
                    > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.2:0::111111111111111111::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:04:53, localpref 100, from 198.51.100.2
                      AS path: I, validation-state: unverified
                    > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2
1:198.51.100.2:11::11111111111111111::1111/304 AD/EVI
                   *[BGP/170] 00:04:53, localpref 100, from 198.51.100.2
                      AS path: I, validation-state: unverified
                    > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2
```

Meaning The outputs show the ESI routes being shared across the VPWS to the remote PE devices.

The routes beginning with **1:198.51.100.x:0::** are the per-Ethernet-segment autodiscovery Type 1 EVPN routes originating from the remote PE devices. The route distinguishers (RDs) are derived at the global level of the devices.

The routes beginning with **1:198.51.100.x:11::** are the per-EVI autodiscovery Type 1 EVPN routes from the remote PE devices. The RDs are taken from the remote PE devices' routing instances.

Verifying Local EVPN Table Route Information

Purpose Verify that the local EVPN routing tables are being populated.

Action Verify that both local and remote reachability information is being added into the EVPN table.

```
user@PE1> show route table EVPN-VPWS.evpn.0

EVPN-VPWS.evpn.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:11::11111111111111111::1111/304 AD/EVI
```

```
                   *[EVPN/170] 00:06:55
                      Indirect
1:198.51.100.3:0::999999999999999999::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:03:24, localpref 100, from 198.51.100.3
                      AS path: I, validation-state: unverified
                    > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.3:11::99999999999999999::9999/304 AD/EVI
                   *[BGP/170] 00:03:25, localpref 100, from 198.51.100.3
                      AS path: I, validation-state: unverified
                    > to 10.0.1.2 via xe-0/0/2.0, label-switched-path PE1toPE3
1:198.51.100.4:0::999999999999999999::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:02:03, localpref 100, from 198.51.100.4
                      AS path: I, validation-state: unverified
                    > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4
1:198.51.100.4:11::99999999999999999::9999/304 AD/EVI
                   *[BGP/170] 00:02:03, localpref 100, from 198.51.100.4
                      AS path: I, validation-state: unverified
                    > to 10.0.2.2 via xe-0/0/3.0, label-switched-path PE1toPE4
```

user@PE3> **show route table EVPN-VPWS.evpn.0**

```
EVPN-VPWS.evpn.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

1:198.51.100.1:0::11111111111111111::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:05:01, localpref 100, from 198.51.100.1
                      AS path: I, validation-state: unverified
                    > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.1:11::111111111111111111::1111/304 AD/EVI
                   *[BGP/170] 00:05:01, localpref 100, from 198.51.100.1
                      AS path: I, validation-state: unverified
                    > to 10.0.1.1 via xe-0/0/0.0, label-switched-path PE3toPE1
1:198.51.100.2:0::11111111111111111::FFFF:FFFF/304 AD/ESI
                   *[BGP/170] 00:05:01, localpref 100, from 198.51.100.2
                      AS path: I, validation-state: unverified
                    > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2
1:198.51.100.2:11::111111111111111111::1111/304 AD/EVI
                   *[BGP/170] 00:05:01, localpref 100, from 198.51.100.2
                      AS path: I, validation-state: unverified
                    > to 10.0.3.1 via xe-0/0/1.0, label-switched-path PE3toPE2
1:198.51.100.3:11::99999999999999999::9999/304 AD/EVI
                   *[EVPN/170] 00:05:25
                      Indirect
```

Meaning In addition to the remote ESI routes being shared across the VPWS, as explained in the previous section, the EVPN table on each PE device also includes a local ESI route. This Type 1 route represents the locally configured Ethernet segment, and is derived from the locally configured RD and ESI values.

Related Documentation

- EVPN Multihoming Overview on page 29
- Overview of VPWS with EVPN Signaling Mechanisms on page 666

- Configuring VPWS with EVPN Signaling Mechanisms on page 681
- vpws-service-id on page 1090
- show evpn vpws-instance on page 1193

PART 5

EVPN-ETREE

CHAPTER 23

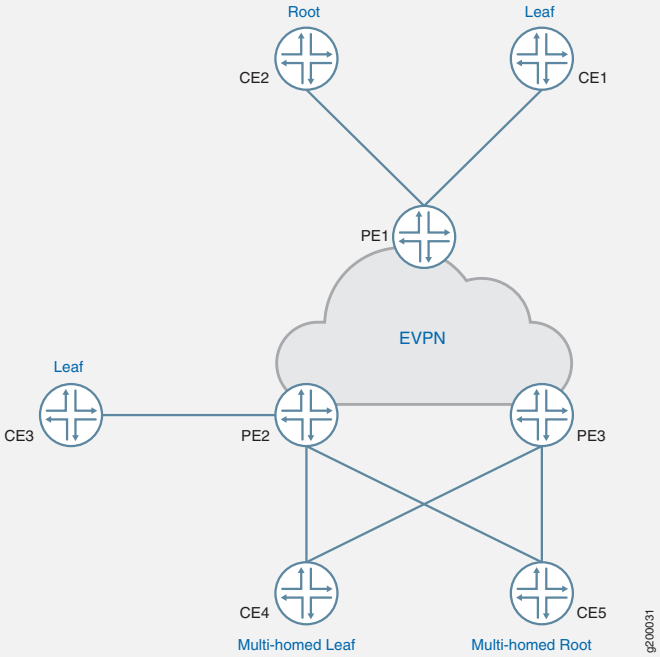
Overview

EVPN-ETREE Overview

The EVPN-ETREE service is a VPN service where each attachment circuit is designated as either root or leaf. The EVPN E-Tree feature implements E-Tree service as defined by the Metro Ethernet Forum (MEF) in draft-sajassi-l2vpn-evpn-etree-03. The E-Tree service is a rooted-multipoint service that is supported only with EVPN over MPLS in the core. The EVPN E-Tree feature provides a way to categorize the interfaces as either “root” or “leaf” in a routing instance. In an EVPN E-Tree service, each Customer Edge devices attached the service is either a root or a leaf. The EVPN E-Tree service adheres to the following forwarding rules:

- A leaf can send or receive traffic only from a root.
- A root can send traffic to another root or any of the leaves.
- A leaf or root can be connected to provider edge (PE) devices in singlehoming mode or multihoming mode.

Figure 69: EVPN E-TREE Service



The EVPN ETREE service has all the benefits of EVPN such as active-active multihoming, load balancing loop detection for E-Tree

In an EVPN ETREE service, the forwarding rule depends on the traffic source and destination. Table 1 shows the forwarding rules within the ETREE service.

Type of Traffic	Allowed/Not-Allowed	Filtering Location
Known Unicast Traffic from Root to Root	Allowed	
Known Unicast Traffic from Root to Leaf	Allowed	
BUM Traffic from Root to Root	Allowed	
BUM Traffic from Root to Leaf	Allowed	
Known Unicast Traffic from Leaf to Leaf	Not Allowed	At the ingress Packet Forwarding Engine
Known Unicast Traffic from Leaf to Root	Allowed	
BUM Traffic from Leaf to Leaf	Not Allowed	At the Egress Packet Forwarding Engine
BUM Traffic from Leaf to Root	Allowed	

If you do not configure a role for an interface, it will be assigned the role of “root” by default. All leaf interfaces are assigned a new mesh group with no local switching set to TRUE. This enables the ingress filtering for unicast traffic and all the leaf-to-leaf traffic will get dropped at ingress leaf interface. For BUM traffic, the filtering will happen at the egress Provider Edge based on the root/leaf label being carried in the packet.

NSR and Unified ISSU Support for EVPN-ETREE

Nonstop active routing (NSR) and graceful Routing Engine switchover (GRES) minimize traffic loss when there is a Routing Engine switchover. When a Routing Engine fails, NSR and GRES enable a routing platform with redundant Routing Engines to switch over from a primary Routing Engine to a backup Routing Engine and continue forwarding packets. Unified in-service software upgrade (ISSU) allows you to upgrade your Junos OS software on your MX Series router with no disruption on the control plane and with minimal disruption of traffic. Both GRES and NSR must be enabled to use unified ISSU.

To enable GRES, include the **graceful-switchover** statement at the **[edit chassis redundancy]** hierarchy level.

Junos OS mirrors essential data when NSR is enabled. For EVPN ETREE, the local EVPN ETREE leaf label that is advertised to other PE as part of the ETREE extended community will be mirrored on the standby Routing Engine. For information on other mirrored data and NSR data flow, see “NSR and Unified ISSU Support for EVPN Overview” on page 235.

To enable NSR, include the **nonstop-routing** statement at the **[edit routing-options]** hierarchy level and the **commit synchronize** statement at the **[edit system]** hierarchy level.

PART 6

Using EVPN for Interconnection

CHAPTER 24

Interconnecting VXLAN Data Centers With EVPN

VXLAN Data Center Interconnect Using EVPN Overview

Starting in Junos OS Release 16.1, Ethernet VPN (EVPN) technology can be used to interconnect Virtual Extensible Local Area Network (VXLAN) networks over an MPLS/IP network to provide data center connectivity. This is done through Layer 2 intra-subnet connectivity and control-plane separation among the interconnected VXLAN networks.

The following sections describe the technology and implementation overview of integrating EVPN with VXLAN to be used as a data center interconnect (DCI) solution.

Technology Overview of VXLAN-EVPN Integration for DCI

The following sections provide a conceptual overview of VXLAN, EVPN, the need for their integration for DCI and the resulting benefits.

Understanding VXLAN

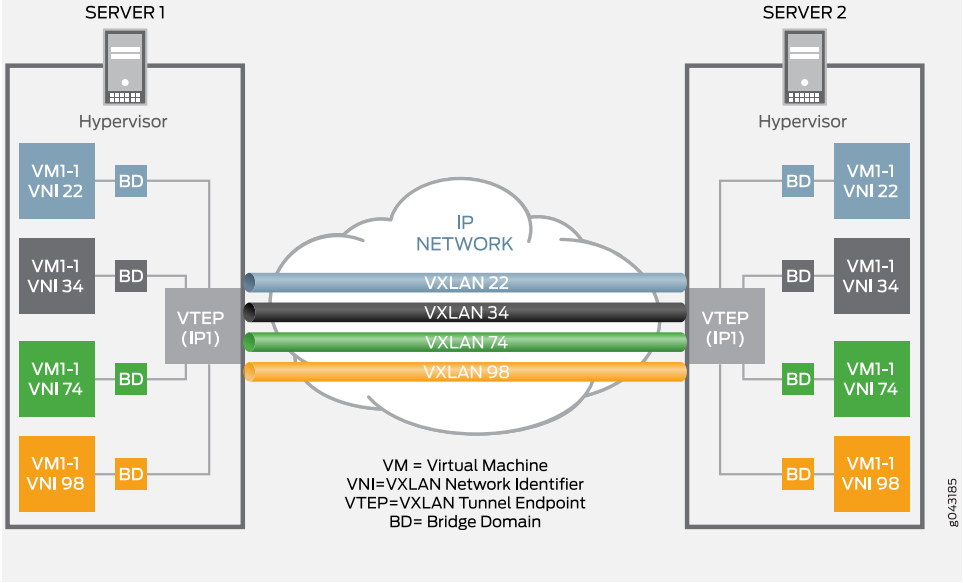
Virtual Extensible Local Area Network (VXLAN) is a Layer 3 encapsulation protocol that enables MX Series routers to push Layer 2 or Layer 3 packets through a VXLAN tunnel to a virtualized data center or the Internet. Communication is established between two virtual tunnel endpoints (VTEPs), which can be end hosts or network switches or routers,

that encapsulate and de-encapsulate the virtual machine (VM) traffic into a VXLAN header.

VXLAN is often described as an overlay technology because it allows you to stretch Layer 2 connections over an intervening Layer 3 network by encapsulating (tunneling) Ethernet frames in a VXLAN packet that includes IP addresses. This feature of VXLAN addresses the requirements of a multi-tenant datacenter, where each tenant's VM might be sharing the physical server with other tenants that are distributed across physical servers within or across different data centers, by meeting the growing need to provide seamless Layer 2 connectivity between all the VMs owned by a tenant, in addition to isolating each tenant's traffic for security and potential MAC address overlaps.

VXLAN tunnels are created between the physical servers by the hypervisors. Since a physical server can be hosting multiple tenants, each hypervisor creates multiple VXLAN tunnels.

Figure 70: VXLAN Overview



VXLAN is a technology that allows you to segment your networks (as VLANs do) but that also solves the scaling limitation of VLANs and provides benefits that VLANs cannot. Some of the important benefits of using VXLANs include:

- You can theoretically create as many as 16 million VXLANs in an administrative domain (as opposed to 4094 VLANs on a Juniper Networks device).

  MX Series routers support as many as 32K VXLANs. This means that VXLANs provide network segmentation at the scale required by cloud builders to support very large numbers of tenants.

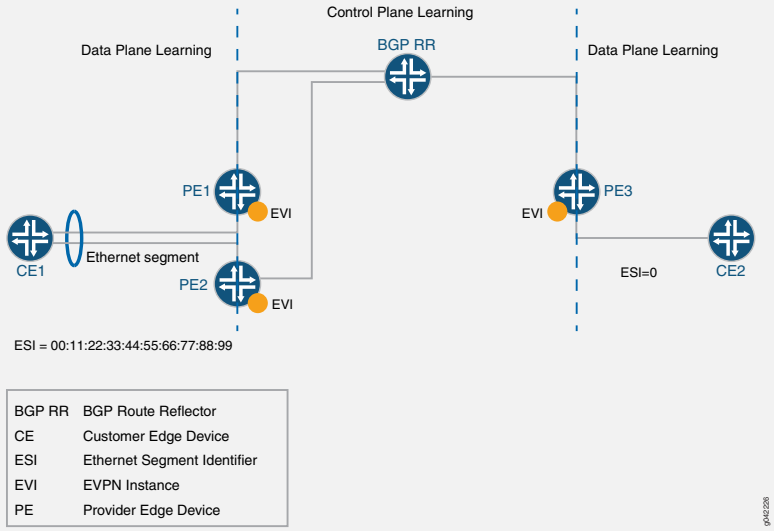
- You can enable migration of virtual machines between servers that exist in separate Layer 2 domains by tunneling the traffic over Layer 3 networks. This functionality allows you to dynamically allocate resources within or between data centers without being

constrained by Layer 2 boundaries or being forced to create large or geographically stretched Layer 2 domains.

Understanding EVPN

EVPN is a new standards-based technology that provides virtual multi-point bridged connectivity between different Layer 2 domains over an IP or IP/MPLS backbone network. Similar to other VPN technologies, such as IPVPN and VPLS, EVPN instances (EVIs) are configured on PE routers to maintain logical service separation between customers. The PEs connect to CE devices which can be a router, switch, or host. The PE routers then exchange reachability information using Multi-Protocol BGP (MP-BGP) and encapsulated traffic is forwarded between PEs. Because elements of the architecture are common with other VPN technologies, EVPN can be seamlessly introduced and integrated into existing service environments.

Figure 71: EVPN Overview



The EVPN technology provides mechanisms for next generation Data Center Interconnect (DCI) by adding extended control plane procedures to exchange the Layer 2 (MAC address) and Layer 3 (IP address) information among the participating Data Center Border Routers (DCBRs). These features help in addressing some of the DCI challenges, such as seamless VM mobility and optimal IP routing. Seamless VM mobility refers to the challenge of Layer 2 extension and maintaining connectivity in the face of VM mobility, and optimal IP routing refers to the challenge of supporting default gateway behavior for a VM's outbound traffic and triangular routing avoidance of a VM's inbound traffic.

The EVPN technology is used by the data center operator to offer multi-tenancy, flexible and resilient services that can be extended on demand. This flexibility and resiliency can require using compute resources among different physical data centers for a single service (Layer 2 extension), and VM motion.

EVPN supports all-active multihoming which allows a CE device to connect to two or more PE routers such that traffic is forwarded using all of the links between the devices. This enables the CE to load balance traffic to the multiple PE routers. More importantly it allows a remote PE to load balance traffic to the multihomed PEs across the core network. This load balancing of traffic flows between data centers is known as aliasing. EVPN also has mechanisms that prevent the looping of broadcast, unknown unicast, and multicast (BUM) traffic in an all-active multi-homed topology.

Multihoming provides redundancy in the event that an access link or one of the PE routers fails. In either case traffic flows from the CE towards the PE use the remaining active links. For traffic in the other direction, the remote PE updates its forwarding table to send traffic to the remaining active PEs connected to the multihomed Ethernet segment. EVPN provides a fast convergence mechanism so that the time it takes to make this adjustment is independent of the number of MAC addresses learned by the PE.

EVPN's MP-BGP control plane allows live virtual machines to be dynamically moved from one data center to another, also known as VM motion. After a VM is moved to a destination server/hypervisor it transmits a Gratuitous ARP which updates the Layer 2 forwarding table of the PE at the destination data center. The PE then transmits a MAC route update to all remote PEs which in turn update their forwarding tables. In this manner, an EVPN tracks the movement of the VM, also known as MAC Mobility. EVPN also has mechanisms to detect and stop MAC flapping.

The EVPN technology, similar to Layer 3 MPLS VPN, is a technology that introduces the concept of routing MAC addresses using MP-BGP over MPLS core. Some of the important benefits of using EVPNs include:

- Ability to have a dual active multihomed edge device.
- Provides load balancing across dual-active links.
- Provides MAC address mobility.
- Provides multi-tenancy.
- Provides aliasing.
- Enables fast convergence.

VXLAN-EVPN Integration Overview

VXLAN defines a tunneling scheme to overlay Layer 2 networks on top of Layer 3 networks. It allows for optimal forwarding of Ethernet frames with support for multipathing of unicast and multicast traffic with the use of UDP/IP encapsulation for tunneling, and is mainly used for the intra-datacenter site connectivity.

On the other hand, a unique characteristic of EVPN is that MAC address learning between PE devices occurs in the control plane. A new MAC address detected from a CE device is advertised by the local PE, using MP-BGP, to all the remote PE devices. This method differs from existing Layer 2 VPN solutions such as VPLS, which learn by flooding unknown unicast in the data plane. This control plane based MAC learning method is the key enabler of the many useful features provided by EVPN.

Because MAC learning is handled in the control plane, this leaves EVPN with the flexibility to support different data plane encapsulation technologies between PEs. This is important because not every backbone network may be running MPLS, especially in Enterprise networks.

There is a lot of interest in EVPN today because it addresses many of the challenges faced by network operators that are building data centers to offer cloud and virtualization services. The main application of EVPN is Data Center Interconnect (DCI), the ability to extend Layer 2 connectivity between different data centers which are deployed to improve the performance of delivering application traffic to end users and for disaster recovery.

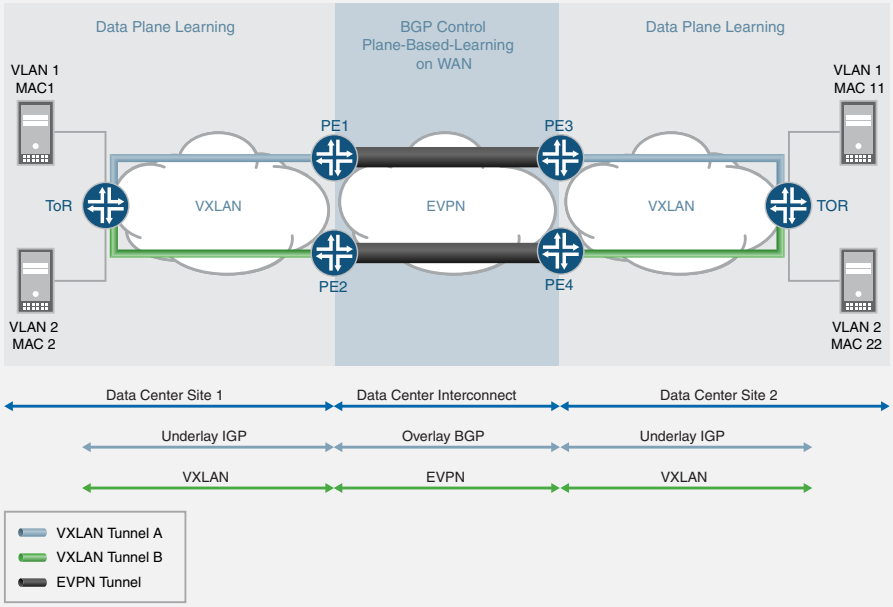
Although there are various DCI technologies available, EVPN has an added advantage over the other MPLS technologies because of its unique features, such as active-active redundancy, aliasing, and mass MAC withdrawal. As a result, to provide a solution for DCI, VXLAN is integrated with EVPN.

Every VXLAN network, which is connected to the MPLS or IP core, runs an independent instance of the IGP control plane. Each PE device participates in the IGP control plane instance of its VXLAN network. Here, each customer is a datacenter so it has its own virtual router for VXLAN underlay.

Each PE node may terminate the VXLAN data plane encapsulation where each VNI or VSID is mapped to a bridge domain. The PE router performs data plane learning on the traffic received from the VXLAN network.

Each PE node implements EVPN to distribute the client MAC addresses learnt over the VXLAN tunnel into BGP. Each PE node encapsulates the VXLAN or Ethernet frames with MPLS when sending the packets over the MPLS core and with the VXLAN tunnel header when sending the packets over the VXLAN network

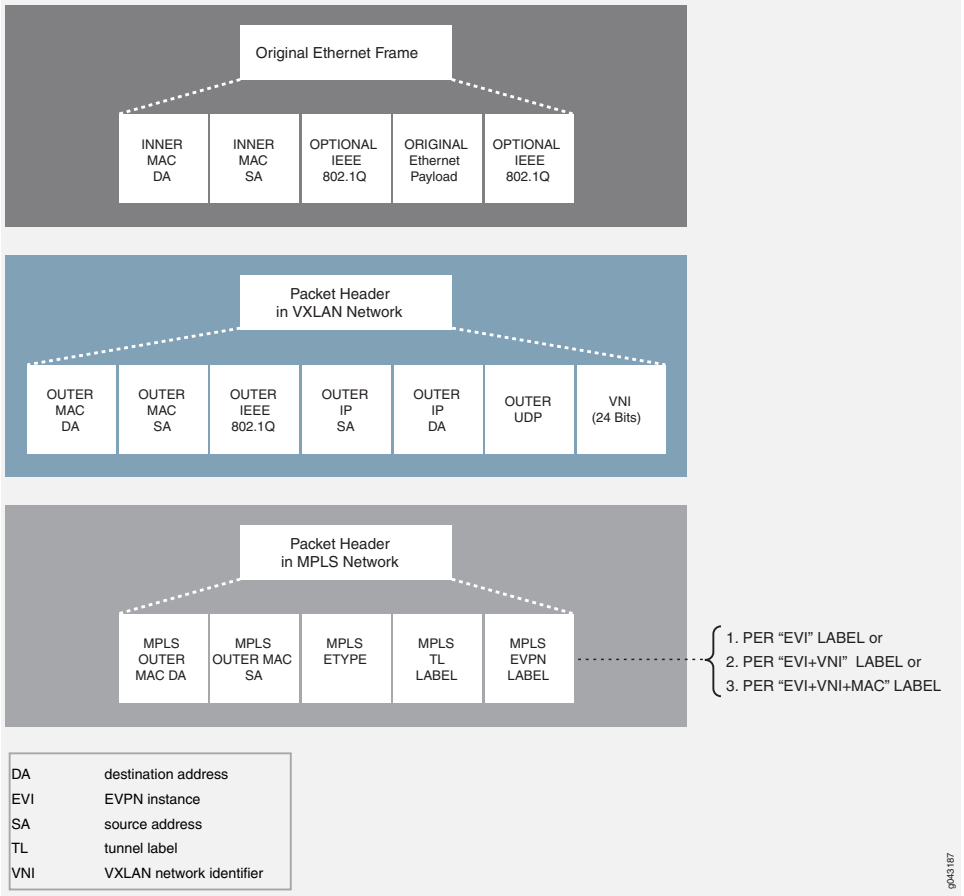
Figure 72: VXLAN-EVPN Integration Overview



VXLAN-EVPN Packet Format

The VXLAN and EVPN packet format is as follows:

Figure 73: VXLAN-EVPN Packet Format



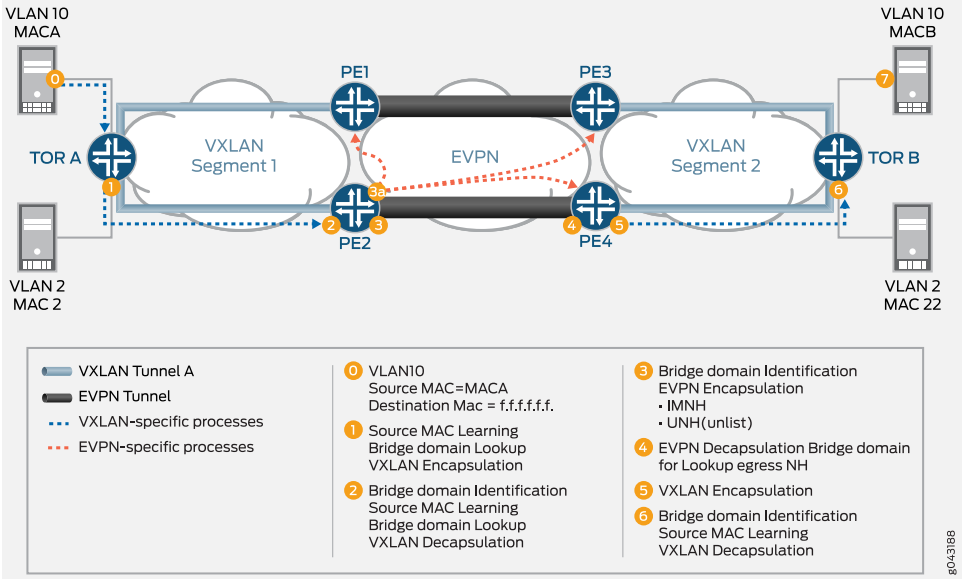
VXLAN-EVPN Packet Walkthrough

The following sections describe the packet walkthrough for two types of traffic between the VXLAN and EVPN networks:

- BUM Traffic Handling on page 718
- Unicast Traffic Handling on page 720

BUM Traffic Handling

Figure 74: VXLAN-EVPN BUM Traffic Handling



The VXLAN to EVPN BUM traffic from VXLAN segment1 to VXLAN segment2 over the EVPN cloud is handled as follows:

1. **0**—On boot up, Server A wants to send traffic to Server B. Because Server A does not have an ARP binding for Server B in its ARP table, Server A builds an ARP broadcast request and sends it. and for ARP.

   The contents of the ARP packets are as follows:

   - VLAN ID = VLAN 10
   - Source MAC= MACA (server A interface MAC)
   - Destination MAC = ff.ff.ff.ff.ff.ff
   - Source IP address = IP address of Server A or VM IP address
   - Destination IP address = IP address of Server B
   - Ether type of packet = 0x0806

   A Layer 2 frame is sent to top-of-rack (TOR) switch TOR A, which is VXLAN enabled.

2. **1**—The ARP request (broadcast) frame is received by switch TOR A. TOR A switch is the originator and terminator of the VXLAN VTEP for VNI 1000. The VTEP for VXLAN 1000 is part of the broadcast domain for Server A VLAN 100.

   After receiving the frame, ingress processing that includes the ingress packet classification is performed by switch TOR A does. Based on the incoming VLAN in the

packet, the packet gets classified into one of the IFL under a given port. The family of this IFL is a bridge family. Based on the IFL bridge family, the bridge domain ID is identified.

After the bridge domain identification, the incoming frame source MAC is learned such that MAC A is reachable through this IFL. Because the frame is a broadcast frame, the frame needs to be sent over to all the members of the broadcast domain (other than the member on which the frame was received). One of the members is the VTEP for VNI 1000. For sending the frame on VXLAN segment, VXLAN BUM next hop processing on the frame is done. The next hop pushes VXLAN header.

The contents of the VXLAN header is as follows:

- Source MAC Address = MAC Address or source IP address interface
- Destination MAC address = Multicast MAC address
- Source IP address = 10.10.10.1
- Destination IP Address = Multicast group address (226.0.39.16)
- Source UDP port = Calculated based on the hash on the incoming frame header
- Destination UDP port = 4789 (well known port for VXLAN tunnel)

After building the VXLAN encapsulated frame, the frame is sent to Router PE2.

3. **2**—Router PE2 receives the VXLAN frame and identifies the frame as a VXLAN frame by looking at the well-known destination UDP port. This VXLAN frame's VNI ID is used for bridge domain identification. After the bridge domain is identified, MAC learning is done for the inner source MAC to the outer source IP address (MACA to 10.10.10.1 mapping). After the mapping is done, the VXLAN header is removed by VDNH.

4. **3A**—After MAC learning is done, the learnt source MAC (MAC1 to outer source IP) is sent to the L2ALD. This MAC route is sent by L2ALD to RPD for control plane learning of this MAC through BGP MAC route advertisement to BGP peers. After the BGP peer routers receive the MAC route advertisement, the routers install this MAC reachability (MACA, MPLS LABEL L1) in the bridge-domain table.

5. **3**—The given bridge domain points to the multicast next hop route for forwarding the packet over the EVPN cloud. This next hop pushes the service label (multicast MPLS label associated with VNI per peer ID, bridge domain, label is the per peer ID and VNI id). The MPLS packet is formed and sent over the MPLS cloud.

6. **4**—The frame is received by Router PE4 as an MPLS packet. Here the bridge domain identification is done by doing theMPLS label L1 lookup in the mpls.0 table. MPLS lookup points to the table next hop for the bridge domain next hop. After the bridge domain is identified and the packet is identified as a broadcast packet, the BUM composite flood next hop is executed. The BUM composite next hop also points to the VXLAN next hop (that is used for building the VXLAN multicast packet).

7. **5**—The VXLAN next hop contains information for building the VXLAN header.

   The VXLAN header information is as follows:

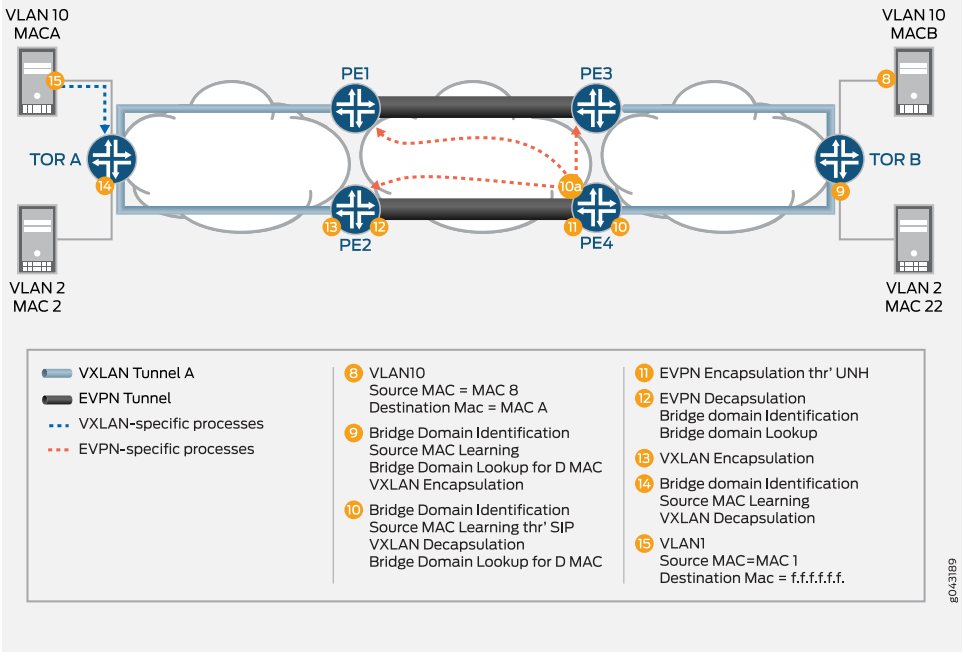
   - Source MAC Address = MAC Address or the source IP address interface
   - Destination MAC address = Multicast MAC Address
   - Source IP address = 11.10.10.1
   - Destination IP Address = Multicast group address (226.0.39.16)
   - Source UDP port = Calculated based on the hash on the incoming frame header
   - Destination UDP port = 4789 (well known port for the VXLAN tunnel)

8. **6**—Frame handling for this step is same as step 1. After the VXLAN header is removed, the frame is forwarded to the CE flood route associated with the broadcast domain, and the packet is forwarded as a Layer 2 frame.

9. **7**—Server B receives an ARP request packet and sends an ARP reply to Server A.

Unicast Traffic Handling

Figure 75: VXLAN-EVPN Unicast Traffic Handling



Assuming that both data and control plane MAC learning has already happened, the VXLAN to EVPN unicast traffic (ARP reply) from Server B is handled as follows:

1. **8**—Server B generates an ARP reply.

   The contents of the ARP packets are as follows:

   - VLAN ID = VLAN 10
   - Source MAC = MACB (Server B interface MAC)
   - Destination MAC = MACA
   - Source IP address = IP address of Server B or VM IP address
   - Destination IP address = IP address of Server A

   The ARP packet is forwarded to switch TOR B.

2. **9**—After receiving the frame, the classification of the incoming frame is done by switch TOR B. The frame is classified in an IFL on the received interface. Based on the IFL family, bridge domain associated with IFL is identified. On the given bridge domain, the source MAC address learning is done. Now the bridge domain destination MAC (MACA) look up is done and this look up provides the VXLAN unicast next hop. This next hop contains all the information about forming the VXLAN header.

   The content of the next hop that is required to form the packet is as follows:

   - Source MAC Address = MAC Address of source IP address interface
   - Destination MAC address = MAC address of the next hop
   - Source IP address = 11.10.10.2
   - Destination IP Address = 11.10.10.1 (as a result of the MAC learning process)
   - Source UDP port = Calculated based on the hash on the incoming frame header
   - Destination UDP port = 4789 (well known port for the VXLAN tunnel)

   NOTE: Earlier version of the VXLAN draft was using 8472 as UDP port.

3. **10**—The VXLAN encapsulated frame is received by Router PE4. At PE4, the frame is identified by doing the lookup using the destination IP address and the destination UDP port. This lookup results in the VXLAN decapsulation. The decapsulation next hop also stores the outer source IP address.

   The next lookup is done based on the VNI ID 1000. This lookup results into the bride domain table.

4. **10A**—Source MAC to source IP address learning is done and the MAC learning notification is received by L2ALD. This MAC is sent to RPD for distribution to other PE routers through BGP-EVPN MAC advertisement route. BGP control plane distributes this MAC reachability information to all other PE routers.

The destination MAC (MAC1) lookup is done in the bridge domain MAC address table. This lookup results into an unicast next hop (EVPN NH).

5. **11**—EVPN unicast nexthop is executed heren. This next hop contains an unicast MPLS service label. This label is distributed through MP-BGP control plane. The downstream peer allocates this MPLS service label. Allocation of this label can be per PE (PE, VLAN) or per MAC address basis. Based on the information in the next hop, MPLS packet is formed and forwarded on the MPLS network.

6. **12**—Router PE2 receives the frame. The frame is identified as an MPLS packet. An MPLS label lookup is done in the MPLS.0 table. This lookup results in the table next hop and in the bridge domain table.

   The destination MAC (MAC1) lookup is done in the bridge domain MAC table. This lookup results in a VXLAN unicast next hop.

7. **13**—VXLAN unicast next hop contain all the information for building the VXLAN encapsulated header. The VXLAN header is imposed on the packet.

   The contents of the VXLAN encapsulation next hop header are as follows:

   - Source MAC Address = MAC Address of ource IP address interface
   - Destination MAC address = MAC address of the next hop
   - Source IP address = 10.10.10.2
   - Destination IP Address = 10.10.10.1 (as a result of the MAC learning process)
   - Source UDP port = Calculated based on the hash on the incoming frame header
   - Destination UDP port = 4789 (well known port for the VXLAN tunnel)

8. **14**—The VXLAN encapsulated frame is received by switch TOR A. At TOR A, the frame is identified by doing the lookup using the destination IP address and the destination UDP port. This lookup results in the VXLAN decapnsulatio. The decapsulated next hop also stores the outer source IP address.

   The next lookup is done based on the VNI ID 1000. This lookup results into the bride domain table. The source MAC (MAC2) to source IP address (10.10.10.2) learning is done. The destination MAC (MAC1) lookup is done in the bridge domain MAC address table. This lookup results into an unicast next hop that has the information about the egress interface.

9. **15**—The ARP reply is received by Server A, and Server A and Server B are ready to communicate.

Implementation Overview of VXLAN-EVPN Integration for DCI

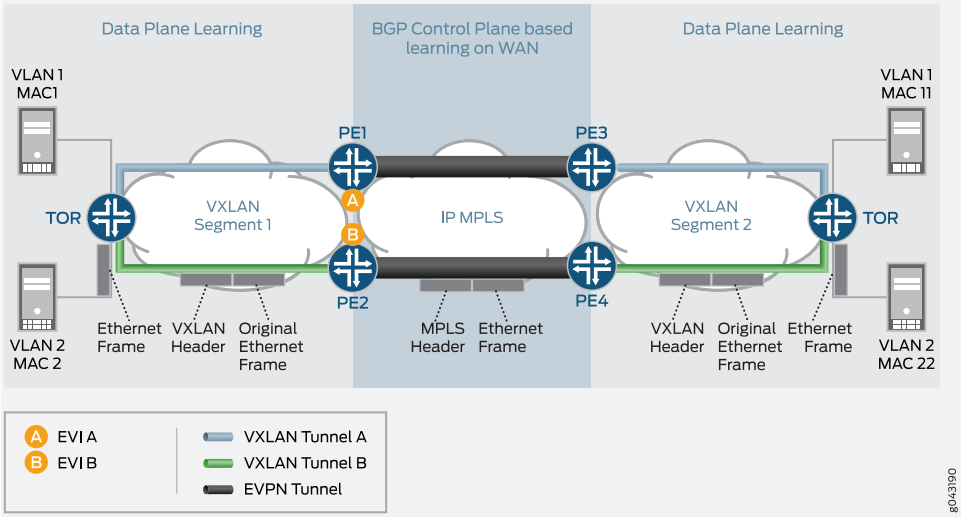
The following sections provide use case scenarios for VXLAN-EVPN integration for DCI.

VNI Base Service Use Case

In the case of the VNI base service, there is one-to-one mapping between a VNI and an EVI. In this case, there is no need to carry the VNI in the MAC advertisement route because the bridge domain ID can be derived from the route-target (RT) associated with this route. The MPLS label allocation is done per EVI basis.

Figure 76 on page 723 provides an overview for VNI base use case scenarios. The VNI base service is used most commonly for achieving the VNI translation and VNI to VLAN interworking.

Figure 76: VNI Base Service

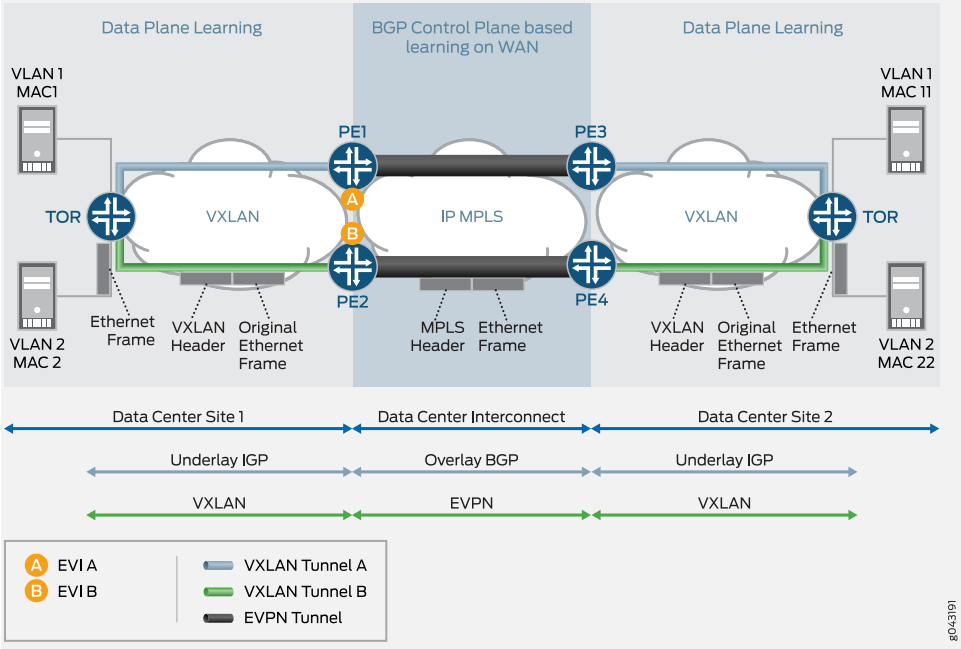


VNI Aware Service Use Case

In the case of VNI aware bundle mode, there are multiple VNIs that can be mapped to the same EVI. The Ethernet tag ID must be set to the VNI ID in the BGP routes advertisements. The MPLS label allocation in this use case should be done per EVI, VNI basis so that the VXLAN can be terminated at the ingress PE router and recreated at the egress PE router.

Figure 77 on page 724 provides details about the VNI aware service used case.

Figure 77: VNI Aware Service

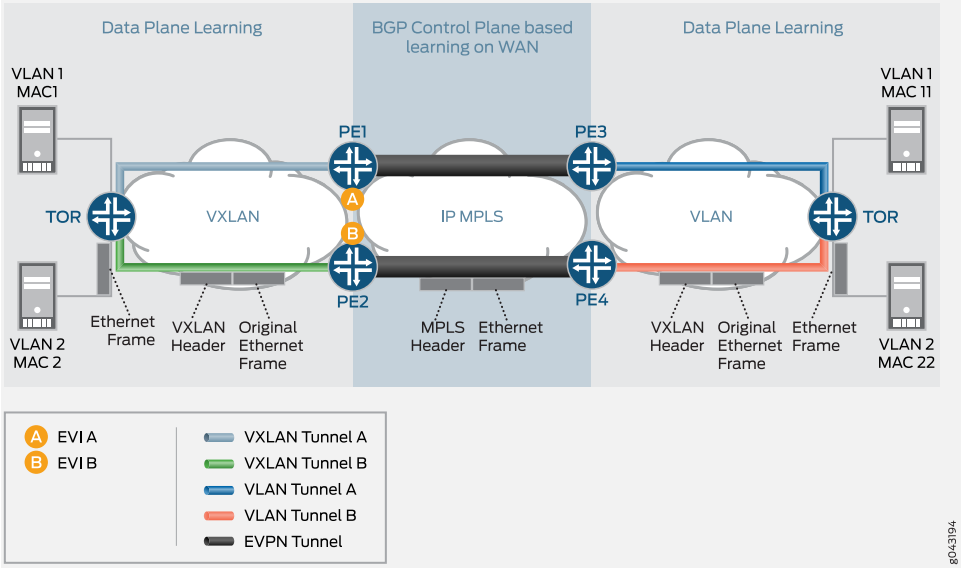


VXLAN-VLAN Interworking Use Case

This use case scenario is required for heterogeneous datacenter sites. In this scenario, the new data center site is a VXLAN-based datacenter site, and the old datacenter sites are based on VLAN. In this scenario, there is a need to have VXLAN interworking with VLAN over EVPN.

Figure 78 on page 725 provides the detail packet walkthrough for the VXLAN-VLAN interworking use case scenario. There is a need to do the VLAN to VXLAN interworking and vice-versa from the control plane BGP route updates perspective. The label allocation needs to be done on a per EVI-basis.

Figure 78: VXLAN-VLAN Interworking

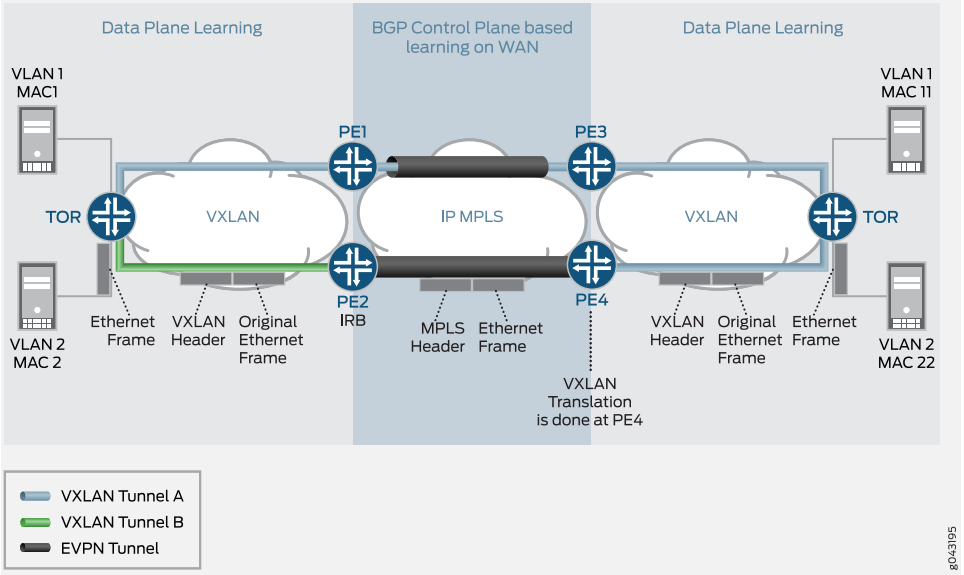


Inter VXLAN Routing Use Case

In this use case, a VM or host in one subnet (VNI-A) want to send traffic to ia VM or host in a different subnet (VNI-B). In order to provide this communication, the inter VXLAN routing should be supported.

Figure 79 on page 726 provides the use case scenarios for the inter VXLAN routing use case.

Figure 79: Inter-VXLAN Routing



Redundancy Use Case

The two types of redundancy use case scenarios include:

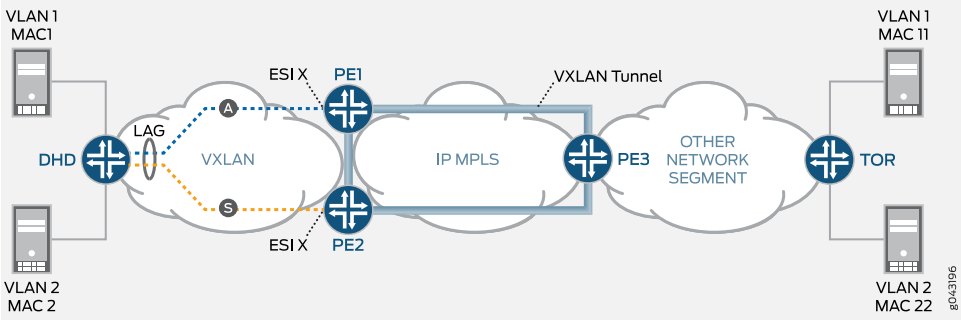
- Active-Standby Redundancy Use Case on page 726

Active-Standby Redundancy Use Case

In this use case scenario, the TOR switch (VXLAN originating GW) or the VXLAN network originating the VXLAN tunnel, is dual homed to two PE devices for active-standby redundancy. If the active link or node fails, then a backup path takes over.

Figure 80 on page 726 provides details of the active-standby redundancy use case scenario.

Figure 80: Active-Standby Redundancy



Supported and Unsupported Features for VXLAN DCI Using EVPN

Junos OS supports the following features for VXLAN DCI using EVPN:

- One-to-one mapping of VXLAN tunnel and an EVPN instance. In other words, one-to-one mapping between a VNI and an EVI.
- Many-to-one mapping of VXLAN tunnels over one EVPN instance, where multiple VNIs can be mapped to the same EVI.
- VNI Translation

NOTE: VNI translation is supported by normalizing a VXLAN tag into a VLAN.

- VXLAN-to-VLAN interworking
- Inter-VXLAN routing
- Single active redundancy
- Active-active redundancy in the PIM BIDIR mode.
- VXLAN tunnel traffic protection using IPSec.
- Graceful Routing Engine switchover.
- ISSU

Junos OS does not support the following functionality for VXLAN DCI using EVPN:

- VXLAN uses IANA assigned UDP port 4789. Packets destined to the UDP port 4789 are processed only when the VXLAN configuration is enabled. The VXLAN packets are decapsulated by the forwarding plane and an inner Layer 2 packet is processed. MAC learnt packets are generated for the control plane processing for newly learnt MAC entries. These entries are throttled using existing infrastructure for MAC learning. VXLAN generates addition learn messages for the remote endpoints. These messages are also throttled using the existing infrastructure for the denial of service detection.
- Packets received on the VXLAN tunnel are processed only if the VXLAN identifier in the packet is a known entity to the device. Unknown entities are discarded by the forwarding plane.
- Using configurable firewall filters can be discarded before it reaches the VXLAN processing module in the forwarding plane of the MX series routers.
- Logical systems

Release History Table

Release	Description
16.1	Starting in Junos OS Release 16.1, Ethernet VPN (EVPN) technology can be used to interconnect Virtual Extensible Local Area Network (VXLAN) networks over an MPLS/IP network to provide data center connectivity.

Example: Configuring VXLAN Data Center Interconnect Using EVPN

This example shows how to configure Virtual Extensible Local Area Network (VXLAN) data center connectivity using Ethernet VPN (EVPN) to leverage the benefits of EVPN as a data center interconnect (DCI) solution.

Requirements

This example uses the following hardware and software components:

- Two provider edge (PE) devices in different data centers (DCs) acting as VXLAN tunnel endpoints (VTEPs).
- Two customer edge (CE) devices.
- Four host devices connected to each PE and CE device.

Before you begin:

- Configure the device interfaces.
- Configure an IGP, such as OSPF, on all the devices.
- Establish a BGP session between the PE devices.
- Configure MPLS and RSVP on the PE devices.
- Configure PIM on the CE devices and in the routing instance of the PE devices.

Overview

VXLAN is a technology that provides intra data center connectivity using a tunneling scheme to stretch Layer 2 connections over an intervening Layer 3 network.

The Ethernet VPN (EVPN) technology, on the other hand, provides a solution for multipoint Layer 2 VPN services with advanced multihoming capabilities, using BGP control plane over MPLS/IP network.

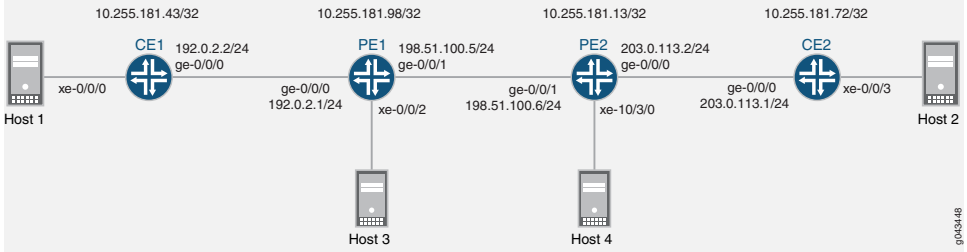
Although several solutions are available for data center connectivity, the integration of EVPN with VXLAN in Junos OS Release 16.1 and later releases, provides an added advantage over the existing MPLS data center interconnect (DCI) technologies.

EVPN provides mechanisms for next generation DCI by adding extended control plane procedures to exchange Layer 2 MAC address and Layer 3 IP address information among the participating Data Center Border Routers (DCBRs). EVPN with its advanced features like active-active redundancy, aliasing, and mass MAC withdrawal helps in addressing the DCI challenges, such as seamless VM mobility and optimal IP routing, thus making it essential to provide VXLAN solutions over EVPN.

Figure 81 on page 729 illustrates VXLAN data center interconnect using EVPN between devices PE1 and PE2 that are located in different data centers (DC1 and DC2, respectively). Each PE device is connected to one CE device and one host. All the PE and CE devices are configured under VLAN 10, and with the same VXLAN Network Identifier (VNI) of 10. Devices CE1 and PE1 belong to the multicast group of 192.168.1.10, and devices CE2 and PE2 belong to the multicast group of 172.16.1.10.

Topology

Figure 81: VXLAN Data Center Interconnect Using EVPN



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter commit from configuration mode.

CE1

```
set interfaces xe-0/0/0 vlan-tagging
set interfaces xe-0/0/0 encapsulation flexible-ethernet-services
set interfaces xe-0/0/0 unit 10 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 10 vlan-id 10
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.181.43/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols pim rp local address 10.255.181.43
set protocols pim interface all
set bridge-domains evpn10 vlan-id 10
set bridge-domains evpn10 interface xe-0/0/0.10
set bridge-domains vxlan vni 10
set bridge-domains vxlan multicast-group 172.16.1.10
set bridge-domains vxlan encapsulate-inner-vlan
```

```
set bridge-domains vxlan decapsulate-accept-inner-vlan
```

CE2

```
set interfaces xe-0/0/3 vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 10 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 10 vlan-id 10
set interfaces lo0 unit 0 family inet address 10.255.181.72/32
set protocols ospf area 0 interface ge-0/0/0.0
set protocols ospf area 0 interface lo0.0 passive
set protocols ospf area 0 interface fxp0.0 disable
set protocols pim rp local address 10.255.181.72
set protocols pim interface all
set bridge-domains evpn10 vlan-id 10
set bridge-domains evpn10 interface xe-0/0/3.10
set bridge-domains vxlan vni 10
set bridge-domains vxlan multicast-group 192.168.1.10
set bridge-domains vxlan encapsulate-inner-vlan
set bridge-domains vxlan decapsulate-accept-inner-vlan
```

PE1

```
set interfaces xe-0/0/2 vlan-tagging
set interfaces xe-0/0/2 encapsulation flexible-ethernet-services
set interfaces xe-0/0/2 unit 10 encapsulation vlan-bridge
set interfaces xe-0/0/2 unit 10 vlan-id 10
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.1/24
set interfaces ge-0/0/1 mtu 1600
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.5/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.255.181.98/32
set protocols rsvp interface all
set protocols mpls no-cspf
set protocols mpls label-switched-path to-PE2 to 10.255.181.13
set protocols mpls interface all
set protocols bgp family evpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.13 local-address 10.255.181.98
set protocols ospf area 0 interface ge-0/0/1.0
set protocols ospf area 0 interface fxp0.0 disable
set protocols ospf area 0 interface lo0.0 passive
set routing-instances evpn10 vtep-source-interface lo0.0
set routing-instances evpn10 instance-type evpn
set routing-instances evpn10 vlan-id 10
set routing-instances evpn10 interface xe-0/0/2.10
set routing-instances evpn10 vxlan vni 10
set routing-instances evpn10 vxlan multicast-group 172.16.1.10
set routing-instances evpn10 vxlan encapsulate-inner-vlan
set routing-instances evpn10 vxlan decapsulate-accept-inner-vlan
set routing-instances evpn10 route-distinguisher 10.255.181.98:10
set routing-instances evpn10 vrf-target target:10:10
set routing-instances evpn10 protocols evpn
set routing-instances evpna instance-type vrf
set routing-instances evpna route-distinguisher 10.255.181.98:11
set routing-instances evpna vrf-target target:65000:11
```

```
set routing-instances evpna vrf-table-label
set routing-instances vrf instance-type vrf
set routing-instances vrf interface ge-0/0/0.0
set routing-instances vrf interface lo0.0
set routing-instances vrf route-distinguisher 10.255.181.98:100
set routing-instances vrf vrf-target target:100:100
set routing-instances vrf protocols ospf area 0 interface lo0.0 passive
set routing-instances vrf protocols ospf area 0 interface ge-0/0/0.0
set routing-instances vrf protocols pim rp static address 10.255.181.43
set routing-instances vrf protocols pim interface all
```

PE2

```
set interfaces ge-0/0/1 mtu 1600
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.6/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces xe-10/3/0 vlan-tagging
set interfaces xe-10/3/0 encapsulation flexible-ethernet-services
set interfaces xe-10/3/0 unit 10 encapsulation vlan-bridge
set interfaces xe-10/3/0 unit 10 vlan-id 10
set interfaces lo0 unit 1 family inet address 10.255.181.13/32
set protocols rsvp interface all
set protocols mpls no-cspf
set protocols mpls label-switched-path to-PE1 to 10.255.181.98
set protocols mpls interface all
set protocols bgp family evpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.98 local-address 10.255.181.13
set protocols ospf area 0 interface ge-0/0/1.0
set protocols ospf area 0 interface fxp0.0 disable
set protocols ospf area 0 interface lo0.0 passive
set routing-instances evpn10 vtep-source-interface lo0.0
set routing-instances evpn10 instance-type evpn
set routing-instances evpn10 vlan-id 10
set routing-instances evpn10 interface xe-10/3/0.10
set routing-instances evpn10 vxlan vni 10
set routing-instances evpn10 vxlan multicast-group 192.168.1.10
set routing-instances evpn10 vxlan encapsulate-inner-vlan
set routing-instances evpn10 vxlan decapsulate-accept-inner-vlan
set routing-instances evpn10 route-distinguisher 10.255.181.13:10
set routing-instances evpn10 vrf-target target:10:10
set routing-instances evpn10 protocols evpn
set routing-instances evpna instance-type vrf
set routing-instances evpna route-distinguisher 10.255.181.13:11
set routing-instances evpna vrf-target target:65000:11
set routing-instances evpna vrf-table-label
set routing-instances vrf instance-type vrf
set routing-instances vrf interface xe-10/3/0.0
set routing-instances vrf interface lo0.0
set routing-instances vrf route-distinguisher 10.255.181.13:100
set routing-instances vrf vrf-target target:100:100
set routing-instances vrf protocols ospf area 0 interface lo0.0 passive
set routing-instances vrf protocols ospf area 0 interface xe-10/3/0.0
set routing-instances vrf protocols pim rp static address 10.255.181.72
set routing-instances vrf protocols pim interface all
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device CE1:

NOTE: Repeat this procedure for Device CE2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Device CE1 interfaces.

```
[edit interfaces]
user@CE1# set xe-0/0/0 vlan-tagging
user@CE1# set xe-0/0/0 encapsulation flexible-ethernet-services
user@CE1# set xe-0/0/0 unit 10 encapsulation vlan-bridge
user@CE1# set xe-0/0/0 unit 10 vlan-id 10
user@CE1# set ge-0/0/0 unit 0 family inet address 192.0.2.2/24
user@CE1# set ge-0/0/0 unit 0 family mpls
user@CE1# set lo0 unit 0 family inet address 10.255.181.43/32
```

2. Enable OSPF on Device CE1 interface, excluding the management interface.

```
[edit protocols]
user@CE1# set ospf area 0.0.0.0 interface ge-0/0/0.0
user@CE1# set ospf area 0.0.0.0 interface lo0.0 passive
user@CE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

3. Enable PIM on all the interfaces of Device CE1.

```
[edit protocols]
user@CE1# set pim rp local address 10.255.181.43
user@CE1# set pim interface all
```

4. Configure an EVPN bridge domain, and assign VLAN ID and interface.

```
[edit bridge-domains]
user@CE1# set evpn10 vlan-id 10
user@CE1# set evpn10 interface xe-0/0/0.10
```

5. Configure a VXLAN bridge domain, assign VXLAN ID, a multicast group address, and encapsulation and decapsulation parameters.

```
[edit bridge-domains]
user@CE1# set vxlan vni 10
user@CE1# set vxlan multicast-group 172.16.1.10
user@CE1# set vxlan encapsulate-inner-vlan
user@CE1# set vxlan decapsulate-accept-inner-vlan
```

Step-by-Step Procedure To configure Device PE1:

NOTE: Repeat this procedure for Device PE2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure Device PE1 interfaces.

```
[edit interfaces]
user@PE1# set xe-0/0/2 vlan-tagging
user@PE1# set xe-0/0/2 encapsulation flexible-ethernet-services
user@PE1# set xe-0/0/2 unit 10 encapsulation vlan-bridge
user@PE1# set xe-0/0/2 unit 10 vlan-id 10
user@PE1# set ge-0/0/0 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-0/0/1 mtu 1600
user@PE1# set ge-0/0/1 unit 0 family inet address 198.51.100.5/24
user@PE1# set ge-0/0/1 unit 0 family mpls
user@PE1# set lo0 unit 1 family inet address 10.255.181.98/32
```

2. Enable MPLS and RSVP on all the interfaces of Device PE1.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set mpls no-cspf
user@PE1# set mpls interface all
```

3. Configure a label-switched-path from Device PE1 to Device PE2.

```
[edit protocols]
user@PE1# set mpls label-switched-path to-PE2 to 10.255.181.13
```

4. Configure internal BGP peering between Devices PE1 and PE2, and enable EVPN signaling for the BGP session.

```
[edit protocols]
user@PE1# set bgp family evpn signaling
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp neighbor 10.255.181.13 local-address 10.255.181.98
```

5. Configure OSPF on Device PE1 interface, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0 interface ge-0/0/1.0
user@PE1# set ospf area 0 interface fxp0.0 disable
user@PE1# set ospf area 0 interface lo0.0 passive
```

6. Configure an EVPN routing instance, assign the VXLAN tunnel endpoint source interface, VLAN ID, assign route distinguisher and VRF target values, and assign Device PE1 interface to the routing instance.

```
[edit routing-instances]
user@PE1# set evpn10 vtep-source-interface lo0.0
user@PE1# set evpn10 instance-type evpn
user@PE1# set evpn10 vlan-id 10
user@PE1# set evpn10 interface xe-0/0/2.10
user@PE1# set evpn10 route-distinguisher 10.255.181.13:10
user@PE1# set evpn10 vrf-target target:10:10
user@PE1# set evpn10 protocols evpn
```

7. Assign the VXLAN ID, multicast group address, and encapsulation and decapsulation parameters for the EVPN routing instance.

```
[edit routing-instances]
user@PE1# set evpn10 vxlan vni 10
user@PE1# set evpn10 vxlan multicast-group 172.16.1.10
user@PE1# set evpn10 vxlan encapsulate-inner-vlan
user@PE1# set evpn10 vxlan decapsulate-accept-inner-vlan
```

8. Configure the first VPN routing and forwarding (VRF) routing instance, and assign route distinguisher and vrf-target values.

```
[edit routing-instances]
user@PE1# set evpna instance-type vrf
user@PE1# set evpna route-distinguisher 10.255.181.13:11
user@PE1# set evpna vrf-target target:65000:11
user@PE1# set evpna vrf-table-label
```

9. Configure the second VRF routing instance, and assign Device PE1 interfaces, route distinguisher and vrf-target values.

```
[edit routing-instances]
user@PE1# set vrf instance-type vrf
user@PE1# set vrf interface ge-0/0/0.0
user@PE1# set vrf interface lo0.0
user@PE1# set vrf route-distinguisher 10.255.181.13:100
user@PE1# set vrf vrf-target target:100:100
```

10. Configure OSPF and PIM protocols for the second VRF routing instance.

```
[edit routing-instances]
user@PE1# set vrf protocols ospf area 0 interface lo0.0 passive
user@PE1# set vrf protocols ospf area 0 interface ge-0/0/0.0
user@PE1# set vrf protocols pim rp static address 10.255.181.43
user@PE1# set vrf protocols pim interface all
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

CE1

```
user@CE1# show interfaces
xe-0/0/0 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
ge-0/0/0 {
    unit 0 {
        family inet {
            address 192.0.2.2/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.181.43/32;
        }
    }
}
user@CE1# show protocols
ospf {
    area 0.0.0.0 {
        interface ge-0/0/0.0;
        interface lo0.0 {
            passive;
        }
        interface fxp0.0 {
            disable;
        }
    }
}
pim {
    rp {
        local {
            address 10.255.181.43;
        }
    }
    interface all;
}
user@CE1# show bridge-domains
evpn10 {
```

```
    vlan-id 10;
    interface xe-0/0/0.10;
    vxlan {
      vni 10;
      multicast-group 172.16.1.10;
      encapsulate-inner-vlan;
      decapsulate-accept-inner-vlan;
    }
}
```

PE1

```
user@PE1# show interfaces
xe-0/0/2 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 10 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
ge-0/0/0 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
    }
}
ge-0/0/1 {
    mtu 1600;
    unit 0 {
        family inet {
            address 198.51.100.5/24;
        }
        family mpls;
    }
}
lo0 {
    unit 1 {
        family inet {
            address 10.255.181.98/32;
        }
    }
}
user@PE1# show protocols
rsvp {
    interface all;
}
mpls {
    no-cspf;
    label-switched-path to-PE2 {
        to 10.255.181.13;
    }
    interface all;
}
```

```
bgp {
   family evpn {
      signaling;
   }
   group ibgp {
      type internal;
      neighbor 10.255.181.13 {
         local-address 10.255.181.98;
      }
   }
}
ospf {
   area 0.0.0.0 {
      interface ge-0/0/1.0;
      interface fxp0.0 {
         disable;
      }
      interface lo0.0 {
         passive;
      }
   }
}
user@PE1# show routing-instances
evpn10 {
   vtep-source-interface lo0.0;
   instance-type evpn;
   vlan-id 10;
   interface xe-0/0/2.10;
   vxlan {
      vni 10;
      multicast-group 172.16.1.10;
      encapsulate-inner-vlan;
      decapsulate-accept-inner-vlan;
   }
   route-distinguisher 10.255.181.13:10;
   vrf-target target:10:10;
   protocols {
      evpn;
   }
}
evpna {
   instance-type vrf;
   route-distinguisher 10.255.181.98:11;
   vrf-target target:65000:11;
   vrf-table-label;
}
vrf {
   instance-type vrf;
   interface ge-0/0/0.0;
   interface lo0.0;
   route-distinguisher 10.255.181.98:100;
   vrf-target target:100:100;
   protocols {
      ospf {
         area 0.0.0.0 {
```

```
            interface lo0.0 {
              passive;
            }
            interface ge-0/0/0.0;
          }
        }
        pim {
          rp {
            static {
              address 10.255.181.43;
            }
          }
          interface all;
        }
      }
    }
```

Verificatiton

Confirm that the configuration is working properly.

- Verifying MAC Learning on page 738
- Verifying PIM Reachability on page 739
- Verifying VXLAN Reachability on page 741

Verifying MAC Learning

Purpose Verify the bridging and EVPN MAC table entries on CE and PE devices.

Action On Device CE1, determine the bridging MAC table entries.

From operational mode, run the **show bridge mac-table** command.

```
user@CE1> show bridge mac-table
```

```
 MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
     O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC)

 Routing instance : default-switch
  Bridging domain : evpn10, VLAN : 10
    MAC                 MAC      Logical          NH     RTR
    address             flags    interface        Index  ID
    00:00:00:00:00:11   D        xe-0/0/0.10
    00:00:00:00:00:22   D        vtep.32769
```

On Device PE1, determine the EVPN MAC table entries.

From operational mode, run the **show evpn mac-table** command.

```
user@PE1> show evpn mac-table

 MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
     O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC)

 Routing instance : evpn10
  Bridging domain : __evpn10__, VLAN : 10
    MAC                MAC      Logical          NH     RTR
    address            flags    interface        Index  ID
    00:00:00:00:00:11  D        vtep.32769
    00:00:00:00:00:22  DC                        1048576 1048576
```

Meaning The bridging and EVPN MAC tables have learned the VLAN configurations.

Verifying PIM Reachability

Purpose Verify that the PIM configuration is working properly on the CE and PE devices.

Action On Device CE1, verify PIM configuration.

From operational mode, run the **show pim rps extensive** command.

```
user@CE1> show pim rps extensive

Instance: PIM.master

address-family INET

RP: 10.255.181.43
Learned via: static configuration
Mode: Sparse
Time Active: 00:06:08
Holdtime: 150
Device Index: 161
Subunit: 32769
Interface: pd-0/2/0.32769
Static RP Override: Off
Group Ranges:
        224.0.0.0/4
Register State for RP:
Group          Source          FirstHop        RP Address      State    Timeout

172.16.1.10      203.1.113.11       203.1.113.11       10.255.181.43   Receive
    171

address-family INET6
```

From operational mode, run the **show pim join extensive** command.

```
user@CE1> show pim join extensive
```

```
Instance: PIM.master Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 172.16.1.10
    Source: *
    RP: 10.255.181.43
    Flags: sparse,rptree,wildcard
    Upstream interface: Local
    Upstream neighbor: Local
    Upstream state: Local RP
    Uptime: 00:06:08
    Downstream neighbors:
        Interface: ge-0/0/0.0 (assert winner)
            192.0.2.1 State: Join Flags: SRW Timeout: 201
            Uptime: 00:05:08 Time since last Join: 00:00:08
            Assert Winner: 192.0.2.2 Metric: 0 Pref: 2147483648 Timeout: 82
        Interface: Pseudo-VXLAN
    Number of downstream interfaces: 2

Group: 172.16.1.10
    Source: 10.255.181.43
    Flags: sparse,spt
    Upstream interface: Local
    Upstream neighbor: Local
    Upstream state: Local Source, Local RP, No Prune to RP
    Keepalive timeout: 338
    Uptime: 00:04:15
    Downstream neighbors:
        Interface: ge-0/0/0.0
            192.0.2.1 State: Join Flags: S Timeout: 201
            Uptime: 00:04:15 Time since last Join: 00:00:08
        Interface: Pseudo-VXLAN
    Number of downstream interfaces: 2

Group: 172.16.1.10
    Source: 203.1.113.11
    Flags: sparse,spt
    Upstream interface: ge-0/0/0.0
    Upstream neighbor: 192.0.2.1 (assert winner)
    Upstream state: Local RP, Join to Source, No Prune to RP
    Keepalive timeout: 338
    Uptime: 00:04:15
    Downstream neighbors:
        Interface: ge-0/0/0.0 (pruned)
            192.0.2.1 State: Prune Flags: SR Timeout: 201
            Uptime: 00:04:15 Time since last Prune: 00:00:08
            Assert Winner: 192.0.2.1 Metric: 0 Pref: 0 Timeout: 179
        Interface: Pseudo-VXLAN
    Number of downstream interfaces: 2

Instance: PIM.master Family: INET6
R = Rendezvous Point Tree, S = Sparse, W = Wildcard
```

Meaning The device reachability using PIM is working as configured.

Verifying VXLAN Reachability

Purpose Verify the connectivity between the VTEPs in the different data centers.

Action From the operational mode, run the **show l2-learning vxlan-tunnel-end-point source**, **show l2-learning vxlan-tunnel-end-point remote**, and **show interfaces vtep** commands.

```
user@PE1> show l2-learning vxlan-tunnel-end-point source

Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
       <default>                0   203.1.113.11        lo0.0    7
    L2-RTT                   Bridge Domain              VNID     MC-Group-IP

    evpn10                   __evpn10__                 10       172.16.1.10
```

```
user@PE2> show l2-learning vxlan-tunnel-end-point source

Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
       <default>                0   203.1.113.12        lo0.0    7
    L2-RTT                   Bridge Domain              VNID     MC-Group-IP
    evpn10                   __evpn10__                 10       192.168.1.10
```

```
user@PE1> show l2-learning vxlan-tunnel-end-point remote

Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
       <default>                0   203.1.113.11        lo0.0    7
 RVTEP-IP         IFL-Idx   NH-Id
 10.255.181.43    2684275660 2684275660
    VNID          MC-Group-IP
    10            172.16.1.10
```

```
user@PE2> show l2-learning vxlan-tunnel-end-point remote

Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
       <default>                0   203.1.113.12        lo0.0    7
 RVTEP-IP         IFL-Idx   NH-Id
 10.255.181.98    351       661
    VNID          MC-Group-IP
    10            192.168.1.10
```

```
user@PE1> show interfaces vtep

Physical interface: vtep, Enabled, Physical link is Up
  Interface index: 133, SNMP ifIndex: 508
  Type: Software-Pseudo, Link-level type: VxLAN-Tunnel-Endpoint, MTU: 1600,
Speed: Unlimited
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Link flags     : None
```

```
  Last flapped   : Never
    Input packets : 0
    Output packets: 0

  Logical interface vtep.32768 (Index 339) (SNMP ifIndex 560)
    Flags: Up SNMP-Traps Encapsulation: ENET2
    Ethernet segment value: 00:00:00:00:00:00:00:00:00:00, Mode: Single-homed,
Multi-homed status: Forwarding
    VXLAN Endpoint Type: Source, VXLAN Endpoint Address: 203.1.113.11, L2 Routing
 Instance: evpn10, L3 Routing Instance: vrf
    Input packets : 0
    Output packets: 0

  Logical interface vtep.32769 (Index 341) (SNMP ifIndex 567)
    Flags: Up SNMP-Traps Encapsulation: ENET2
    VXLAN Endpoint Type: Remote, VXLAN Endpoint Address: 10.255.181.43, L2 Routing
 Instance: evpn10, L3 Routing Instance: vrf
    Input packets : 143746
    Output packets: 95828
    Protocol bridge, MTU: 1600
      Flags: Trunk-Mode
```

Meaning The output shows the correct tunnel source IP address (assigned to the loopback interface), VLAN, and multicast group for the VXLAN. Device PE1 is reachable because its IP address (the address assigned to the loopback interface) appears in the output. The output also shows that the VXLAN (VNI 10) and corresponding multicast group are configured correctly on the remote VTEP, Device PE2.

Related Documentation

- VXLAN Data Center Interconnect Using EVPN Overview on page 711

CHAPTER 25

Interconnecting EVPN-VXLAN Data Centers Through an EVPN-MPLS WAN

EVPN-VXLAN Data Center Interconnect Through EVPN-MPLS WAN Overview

You can interconnect different data center networks running Ethernet VPN (EVPN) with Virtual extensible LAN (VXLAN) encapsulation through a WAN running MPLS-based EVPN.

The following sections describe the technology and implementation overview of interconnecting data center networks running EVPN-VXLAN through a WAN running EVPN-MPLS to be used as a data center interconnect (DCI) solution.

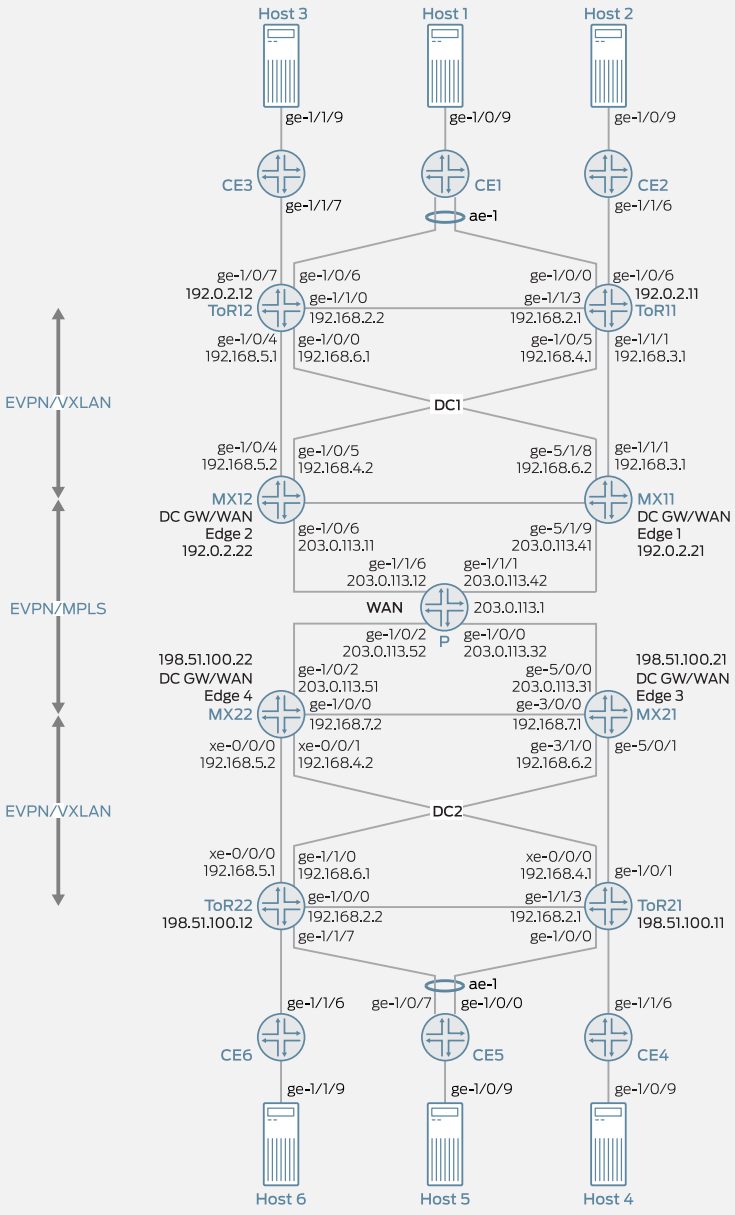
Interconnection of Data Center Networks Through WAN Overview

The following provides a conceptual overview of interconnecting different data center networks running Ethernet VPN (EVPN) with Virtual extensible LAN (VXLAN) encapsulation through a WAN running MPLS-based EVPN using the logical tunnel (lt-) interface. You can:

- Connect data center edge routers over a WAN network running MPLS-based EVPN to achieve data center interconnection.
- Interconnect EVPN-VXLAN and EVPN-MPLS using the logical tunnel (lt-) interface configured on the data center edge routers.

Figure 82 on page 744 illustration shows the interconnection of two data center networks (DC1 and DC2) running EVPN-VXLAN encapsulation through a WAN running MPLS-based EVPN:

Figure 82: EVPN-VXLAN Data Center Interconnect Through WAN Running EVPN-MPLS



In this illustration,

- The following devices are a part of the data center EVPN-VXLAN overlay network 1 (DC1):

- Customer edge devices (CE1, CE2, and CE3) connected to the data center network.
- VLAN hosts connected to each CE device.
- MX routers playing the role of top-of-rack (ToR11 and ToR12) routers.
- MX routers playing the role of the data center gateway router in the EVPN-VXLAN network and as a WAN edge router running MPLS-based EVPN (MX11 and MX12).

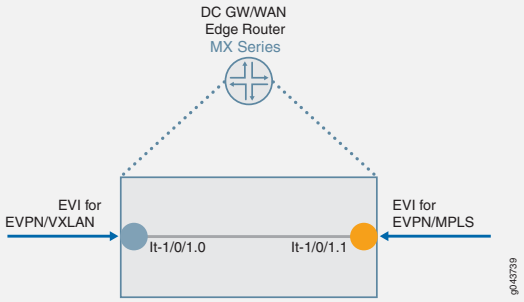
- The following devices are a part of the data center EVPN-VXLAN overlay network 2 (DC2):
  - Customer edge devices (CE4, CE5, and CE6) connected to the data center network.
  - VLAN hosts connected to each CE device.
  - MX routers playing the role of top-of-rack (ToR21 and ToR22) routers.
  - MX routers playing the role of the data center gateway router in the EVPN-VXLAN network and as a WAN edge router running MPLS-based EVPN (MX21 and MX22).

The interconnection of the data center network is realized on the data center gateway router through a pair of logical tunnel (lt-) interface.

On the data center gateway router, you need to configure a pair of logical tunnel (lt-) interface to interconnect the data center EVPN-VXLAN instance and the WAN MPLS-based EVPN instance: one logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN network and the other logical tunnel (lt-) interface as the access interface for MPLS-based EVPN network as shown in the Figure 83 on page 745.

The support for active-active multi-homing is provided at the data center gateway routers for interconnection.

Figure 83: Logical Tunnel (lt-) Interface of DC GW/WAN Edge Router Configured to Interconnect EVPN-VXLAN and EVPN-MPLS Instances



To configure EVPN-VXLAN and MPLS-based EVPN instances on the logical tunnel (lt-) interface of the data center gateway router, see "Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS" on page 752.

Multi-homing on Data Center Gateways

You can configure redundant data center gateways and active-active multi-homing of EVPN-VXLAN network to a WAN running MPLS-based EVPN and active-active multi-homing of MPLS-based EVPN network to EVPN-VXLAN. This allows redundancy between the interconnection of EVPN-VXLAN network and MPLS-based EVPN WAN network. This also enables load-balancing of unicast traffic among the redundant data center gateways in both directions (from EVPN-VXLAN to EVPN-MPLS, as well as from EVPN-MPLS to EVPN-VXLAN). Broadcast, unknown unicast, and multicast (BUM) traffic is forwarded out of the data center by one of the data center gateways.

EVPN Designated Forwarder (DF) Election

To achieve active-active EVPN-VXLAN to EVPN-MPLS interconnect instance and active-active EVPN-MPLS to EVPN-VXLAN instance, the logical tunnel (lt-) interface on the data center gateway router is configured with a non-zero Ethernet Segment Identifier (ESI). The ESI, a 10-octet value that must be unique across the entire network, is configured on a per port basis for the logical tunnel (lt-) interface. As per the EVPN multi-homing procedure defined in the RFC7432, the following routes are advertised for an EVPN instance (EVI):

- Advertise an Ethernet segment route
- Advertise an ESI auto-discovery route with a valid split-horizon label and mode set to multi-homing

The standard EVPN DF election procedure described in RFC7432 is considered. The DF election is based on per Ethernet segment for each EVI. The EVPN-VXLAN and EVPN-MPLS run its DF election process independently.

Split Horizon

Split horizon prevents the looping of BUM traffic in a network, see RFC7432. For BUM traffic from core to the data center gateway (EVPN PE) direction, DF floods the BUM traffic to the access (lt- interface) router and non-DF blocks the BUM traffic. When a DF or non-DF receives the BUM traffic coming from access (lt- interface) router, it gets flooded to the core, but DF does not flood BUM traffic received from its non-DF to the access router based on split horizon rules. For a given BUM packet, only one copy is flooded to the access router (lt- interface) and then to the EVPN core through one of the data center gateway routers because EVPN is multi-homed to another EVPN network. The DF filter rule from the first EVPN instance guarantees that only one copy of BUM traffic is forwarded from the DF to the lt-interface before it re-enters the second EVPN instance.

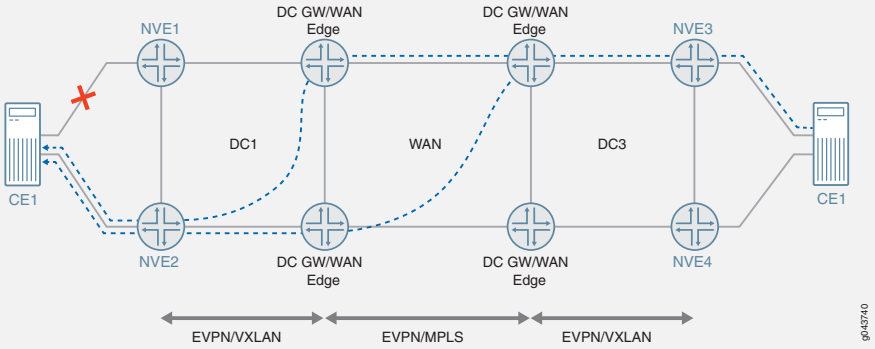
Aliasing

When redundancy is configured in data center gateways, the traffic is load-balanced among redundant data center gateway routers on a per flow basis. MAC address is learned through the data plane using a pair of logical tunnel (lt-) interfaces configured for EVPN-VXLAN instance and EVPN-MPLS instance for data center interconnect. However, the MAC owned by a host is always accessible by all the redundant data center

gateways due to the nature of active-active multi-homing and full-mesh of EVPN PEs in both EVPN-VXLAN network and in WAN running EVPN-MPLS. Each EVPN instance on the data center gateway router declares the support of aliasing function for the ESI configured on the logical tunnel (lt-) interface by advertising per EVI auto-discovery route. The aliasing functionality support is defined in the RFC7432.

Figure 84 on page 747 illustrates a link failure between CE1 and NVE1 but CE1 is still reachable by both data center gateway routers within the data center network (DC1).

Figure 84: Load Balancing Among Redundant DC GW/WAN Edge Routers



A link failure between a host and its top-of-rack (TOR) devices does not impact the aliasing functionality declared by the data center gateway router as the data center network itself is active-active to the WAN running EVPN-MPLS. As long as the host is connected to another ToR device in the data center network, the host is still accessible by all the other redundant data center gateway routers, so the aliasing functionality applies.

VLAN-Aware Bundle Service

In Junos OS for MX Series, both EVPN-VXLAN and EVPN-MPLS instances support VLAN-aware bundle service with one or more bridge domains. To connect two EVIs with VLAN-aware bundle service through a pair of logical tunnel (lt-) interface, it needs trunk interface support on the logical tunnel (lt-) interface, as well as trunk interface support for both EVPN-VXLAN and EVPN-MPLS instances. A truck interface on Junos OS MX Series allows a logical interface to accept packets tagged with any VLAN ID specified in a VLAN ID list.

When the trunk mode is used for the logical tunnel (lt-) interface, the frames going out of the logical tunnel (lt-) interface trunk port from the first EVPN virtual switch are tagged with the appropriate VLAN tag; going through its peer logical tunnel (lt-) interface, the incoming frames to the second virtual switch are inspected and forwarded based on the VLAN tag found within the frame.

The following is a sample configuration to use trunk mode on logical tunnel (lt-) interface to support VLAN-aware bundle service for interconnection of EVPN-VXLAN with a WAN running MPLS-based EVPN:

```
interfaces lt-1/0/10 {
esi {
   36:36:36:36:36:36:36:36:36:36;
   all-active;
}
unit 3 {
   encapsulation ethernet-bridge;
      peer-unit 4;
      family bridge {
      interface-mode trunk;
      vlan-id-list [ 51 52 ];
   }
}
unit 4 {
   encapsulation ethernet-bridge;
      peer-unit 3;
      family bridge {
      interface-mode trunk;
      vlan-id-list [ 51 52 ];
   }
}
}
```

The following is a sample configuration of trunk port support for EVPN-VXLAN and EVPN-MPLS:

```
routing-instances evpn-mpls {
vtep-source-interface lo0.0;
   instance-type virtual-switch;
   interface lt-1/0/10.4;
   route-distinguisher 101:2;
   vrf-target target:2:2;
   protocols {
   evpn {
      extended-vlan-list 51-52;
   }
}
bridge-domains {
   bd1 {
      domain-type bridge;
      vlan-id 51;
   }
   bd2 {
      domain-type bridge;
      vlan-id 52;
   }
}
}
routing-instances red {
vtep-source-interface lo0.0;
   instance-type virtual-switch;
   interface lt-1/0/10.4
   route-distinguisher 101:1;
   vrf-target target:1:1;
```

```
    protocols {
    evpn {
      encapsulation vxlan;
      extended-vni-list all;
    }
}
bridge-domains {
    bd1 {
      domain-type bridge;
        vlan-id 51;
        routing-interface irb.0;
        vxlan {
        vni 51;
          encapsulate-inner-vlan;
          decapsulate-accept-inner-vlan;
        ingress-node-replication;
      }
    }
    bd2 {
      domain-type bridge;
        vlan-id 52;
        routing-interface irb.1;
        vxlan {
        vni 52;
          encapsulate-inner-vlan;
          decapsulate-accept-inner-vlan;
        ingress-node-replication;
      }
    }
}
}
```

Data Center Network Design and Considerations

Before designing a data center network, you need to decide whether to use IGP, or iBGP, or eBGP protocols in the data center network for IP underlay. Another important factor to consider is the AS assignment. The ToR device in the data center network is required to have an AS number that is different than the AS number used in the WAN edge router.

For the overlay network, you need to decide whether to use iBGP, or eBGP, or a combination of both iBGP and eBGP.

Figure 85: Data Center Network Design

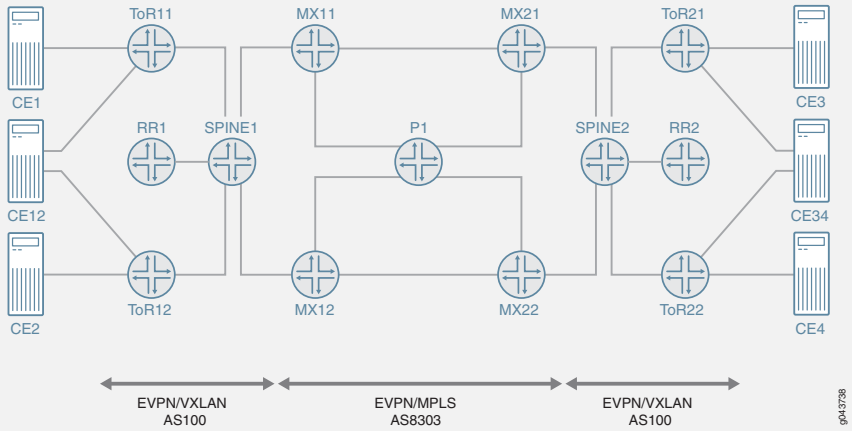


Figure 85 on page 750 illustrates the MX routers (MX11, MX12, MX21 and MX22) as the data center gateway and WAN edge routers that interconnects EVPN-VXLAN to EVPN-MPLS. Spine switches provide connection for east and west traffic among ToRs so that traffic that does not need to be Layer 3 routed does not go through the MX routers. From a network design perspective, to provide an end-to-end EVPN solution, the following requirements must be met:

- Isolate IGP Between EVPN-VXLAN and EVPN-MPLS Segments on page 750
- Different Autonomous Systems (AS) in EVPN-VXLAN and EVPN-MPLS Network on page 751

Isolate IGP Between EVPN-VXLAN and EVPN-MPLS Segments

When IGP is used in the data center network, you need to isolate the IP network in EVPN-VXLAN from the IP network in the WAN. When IGP is used in the data center, one option is not to run IGP protocol on the interfaces that connects spine switches and MX routers. Instead, an eBGP session with address family inet unicast is used between spine switches and MX routers such that through IGP/eBGP/policy you can leak loopback addresses of ToR and MX routers to each other and still maintain the isolation of IP network in the data center from WAN. In the EVPN-VXLAN segment, IGP is between spine switches and ToRs only. In the EVPN-MPLS segment, IGP is between all the MX routers.

- Using iBGP for IP Underlay in the Data Center Network on page 750
- Using eBGP for the IP Underlay in the Data Center Network on page 751

Using iBGP for IP Underlay in the Data Center Network

If the requirement is to not use IGP in the IP underlay in the data center, iBGP with address family inet unicast can be used to replace OSPF between spine switches and ToRs. Between spine switches and data center gateways, you still need to use eBGP for advertising loopback IP.

Using eBGP for the IP Underlay in the Data Center Network

If the requirement is to use eBGP only in the data center, you need to use eBGP with address family inet unicast for the IP underlay. In this case, it is a typical 2 stage CLOS network without spine aggregation layer. Each ToR and data center gateway is assigned a unique AS number. ToR establishes eBGP session with data center gateway routers directly.

Different Autonomous Systems (AS) in EVPN-VXLAN and EVPN-MPLS Network

The following is support for different AS in EVPN-VXLAN and EVPN-MPLS network running iBGP, or eBGP for the IP overlay.

- Runing iBGP/eBGP for the Overlay on page 751
- Running eBGP only for the Overlay on page 751

Runing iBGP/eBGP for the Overlay

ToRs and spine switches are in the same AS100 and all MX Series routers are in AS8303. Among ToRs, its EVPN Network Layer Reachability Information (NLRI) is exchanged through iBGP session. A BGP route reflector (RR) is used and each ToR establishes iBGP session with the RR. Data traffic between ToRs that belongs to the same bridge domain goes through spine switch only and it is always 2 hops away. Since the ToRs and spine switches are in the same AS and the MX edge routers are in the different AS, the MX edge router establishes eBGP session to either RR or each ToR directly. By default, route learned from iBGP session (ToRs) are re-advertised to the eBGP (MX routers) and vice versa. BGP next-hop unchanged is enforced when BGP re-advertises EVPN NLRI among iBGP and eBGP session.

Running eBGP only for the Overlay

If the requirement is to run eBGP only in the data center, each ToR is assigned a unique AS number. Each data center gateway router uses a unique AS number on the data center facing side. For the WAN facing side, the same AS number is used, but the AS number would be different from the AS number used for the data center facing side. The AS number may also be reused in each data center.

To prevent an EVPN route in the data center to be advertised to the data center gateway routers in another data center, you must turn on the route constrain in the EVPN-MPLS network. To make BGP route constrain to work, different route target is used for EVPN-VXLAN and EVPN-MPLS network, respectively.

Related Documentation

- Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS on page 752

Example: Interconnecting EVPN-VXLAN Data Center Networks Through a WAN Running EVPN-based MPLS

This example shows how to interconnect EVPN-VXLAN data center networks through a WAN running EVPN-MPLS to leverage the benefits of EVPN as a Data Center Interconnect (DCI) solution.

Requirements

This example uses the following hardware and software components:

- Four Juniper Networks MX Series routers to be configured as data center gateways and WAN edge routers.
- Four Juniper Networks MX Series routers to be configured as top-of-rack (ToR) routers.
- Six customer edge (CE) devices.
- Six host devices connected to each CE device that has the capability to configure multiple VLANs.
- One provider (P) router part of the EVPN-MPLS WAN network.
- Junos OS Release 17.2, or later.

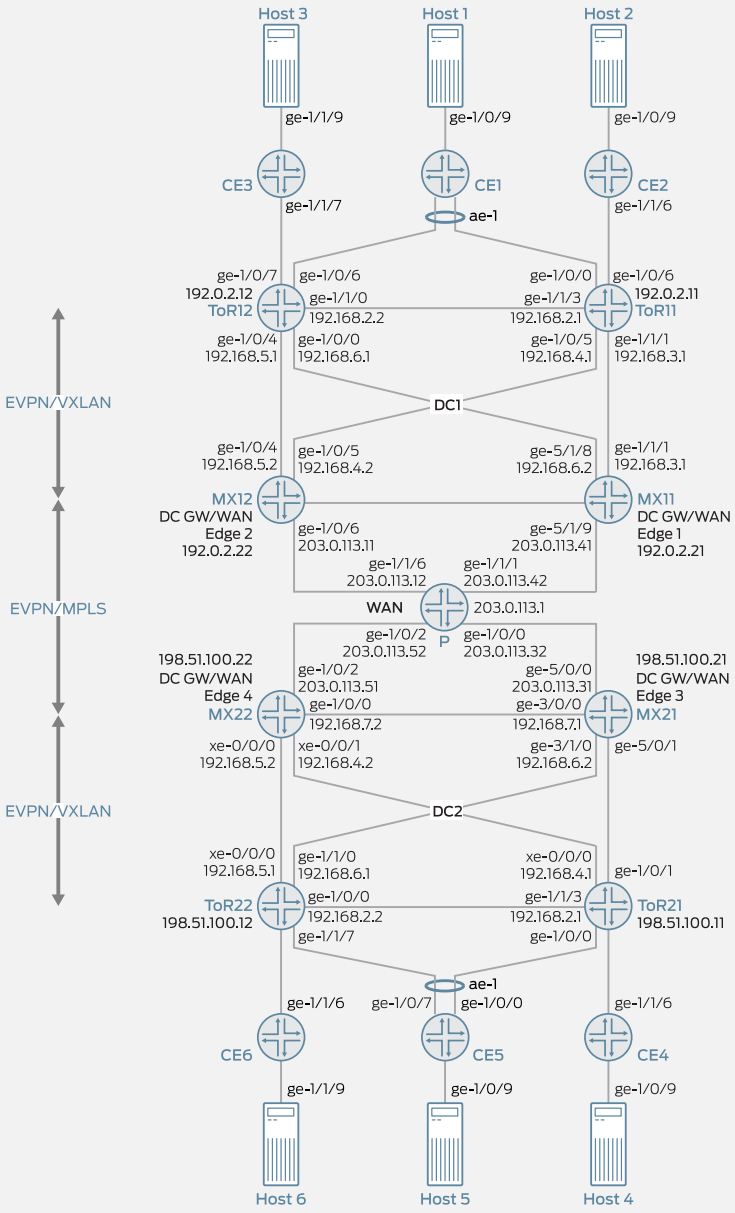
Overview

You can interconnect different data center networks running Ethernet VPN (EVPN) with Virtual extensible LAN (VXLAN) encapsulation through a WAN running MPLS-based EVPN using the logical tunnel (lt-) interface.

Figure 86 on page 753 illustrates the interconnection of data center networks running EVPN with VXLAN encapsulation through a WAN running MPLS-based EVPN. For the purposes of this example, the MX Series routers acting as data center gateways and as WAN edge routers are named MX11, MX12, MX21, and MX22. The MX Series routers acting as top-of-rack (ToR) routers are named ToR11, ToR12, ToR21, and ToR22. The customer edge (CE) devices connected to the data center network 1 (DC1) are named CE1, CE2, and CE3. The customer edge (CE) devices connected to the data center network 2 (DC2) are named CE4, CE5, and CE6. The host devices connected to each CE device should be able to configure multiple host VLANs. The WAN provider router is named as P.

NOTE: CE devices are part of the logical system of ToR devices.

Figure 86: EVPN-VXLAN Data Center Interconnect Through WAN Running EVPN-MPLS



For the MX Series routers acting as data center gateways and WAN edge routers, configure the following information:

- IRB interfaces, virtual gateway addresses, and loopback logical interfaces.
- Multiprotocol external BGP (MP-EBGP) underlay connectivity between gateway and ToR routers, EVPN as the signaling protocol.
- Routing policies to allow specific routes into the virtual-switch tables.

- Routing instances (Layer 3 VRFs) for each virtual network, including a unique route distinguisher, and a vrf-target value.
- Virtual-switch instances (Layer 2 MAC-VRFs) for each virtual network, the VTEP source interface (always lo0.0), route distinguisher, and vrf-import policy.
- EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method for each virtual switch.
- Bridge domain within each virtual switch that maps VNIDs to VLAN IDs, an IRB (Layer 3) interface, and the BUM forwarding method.

For the MX Series routers acting as top-of-rack (ToR) routers, configure the following information:

- Host facing interfaces with VLANs, VLAN IDs, and loopback logical interfaces.
- Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG), Ethernet Segment ID (ESI), and all-active mode.
- Multiprotocol external BGP (MP-EBGP) overlays between ToR and gateway routers using EVPN as the signaling protocol.
- EVPN with VXLAN as the encapsulation method, extended-vni-list, multicast mode, and route targets for each VNI.
- Vrf-imp policy, vtep-source-interface, route-distinguisher, and vrf import and target information.
- VLANs, with VLAN IDs mapped to globally significant VNIs, and VXLAN ingress node replication.

NOTE: You can set the virtual gateway address as the default IPv4 or IPv6 gateway address for end hosts (virtual machines or servers).

Configuration

- Configuring ToR11 on page 780
- Configuring ToR12 on page 786
- Configuring Data Center Gateway and WAN Edge 1 Router (MX11) on page 793
- Configuring Data Center Gateway and WAN Edge 2 Router (MX12) on page 801
- Configuring Data Center Gateway and WAN Edge 3 Router (MX21) on page 812
- Configuring Data Center Gateway and WAN Edge 4 Router (MX22) on page 821
- Configuring ToR21 on page 829
- Configuring ToR22 on page 835

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

ToR11

```
set system host-name ToR11
set logical-systems CE-2 interfaces ge-1/0/9 unit 0 description "CONNECTED TO
Host-2"
set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-2 interfaces ge-1/1/6 unit 0 description "CONNECTED TO
ToR11"
set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-2 bridge-domains BD-1 domain-type bridge
set logical-systems CE-2 bridge-domains BD-1 vlan-id 1
set logical-systems CE-2 bridge-domains BD-2 domain-type bridge
set logical-systems CE-2 bridge-domains BD-2 vlan-id 2
set logical-systems CE-2 bridge-domains BD-3 domain-type bridge
set logical-systems CE-2 bridge-domains BD-3 vlan-id 3
set logical-systems CE-2 bridge-domains BD-4 domain-type bridge
set logical-systems CE-2 bridge-domains BD-4 vlan-id 4
set logical-systems CE-2 bridge-domains BD-5 domain-type bridge
set logical-systems CE-2 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 1
set interfaces traceoptions file RO-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces ge-1/0/5 unit 0 description "CONNECTED TO MX-12"
set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.1/24
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-2"
set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
set interfaces ge-1/1/0 description "CONNECTED TO CE-1"
set interfaces ge-1/1/0 gigether-options 802.3ad ae0
set interfaces ge-1/1/1 unit 0 description "CONNECTED TO MX-11"
set interfaces ge-1/1/1 unit 0 family inet address 192.168.3.1/24
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR12"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.1/24
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
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 11:11:11:11:11:11
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces lo0 unit 81 family inet address 192.0.2.11/32
set routing-options router-id 192.0.2.11
set routing-options autonomous-system 100
set routing-options forwarding-table export evpn-pplb
set protocols bgp local-as 100
set protocols bgp group MX11 type external
set protocols bgp group MX11 local-address 192.168.3.1
set protocols bgp group MX11 export LO
set protocols bgp group MX11 export TEST
set protocols bgp group MX11 peer-as 400
set protocols bgp group MX11 neighbor 192.168.3.2 family inet unicast
```

```
set protocols bgp group MX12 type external
set protocols bgp group MX12 local-address 192.168.4.1
set protocols bgp group MX12 export LO
set protocols bgp group MX12 export TEST
set protocols bgp group MX12 peer-as 500
set protocols bgp group MX12 neighbor 192.168.4.2 family inet unicast
set protocols bgp group ToR12 type external
set protocols bgp group ToR12 local-address 192.168.2.1
set protocols bgp group ToR12 export LO
set protocols bgp group ToR12 export TEST
set protocols bgp group ToR12 peer-as 200
set protocols bgp group ToR12 local-as 100
set protocols bgp group ToR12 neighbor 192.168.2.2 family inet unicast
set protocols bgp group MX11-EVPN type external
set protocols bgp group MX11-EVPN multihop ttl 2
set protocols bgp group MX11-EVPN multihop no-nexthop-change
set protocols bgp group MX11-EVPN local-address 192.0.2.11
set protocols bgp group MX11-EVPN export TEST
set protocols bgp group MX11-EVPN peer-as 400
set protocols bgp group MX11-EVPN local-as 100
set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling
set protocols bgp group MX12-EVPN type external
set protocols bgp group MX12-EVPN multihop ttl 2
set protocols bgp group MX12-EVPN multihop no-nexthop-change
set protocols bgp group MX12-EVPN local-address 192.0.2.11
set protocols bgp group MX12-EVPN export TEST
set protocols bgp group MX12-EVPN peer-as 500
set protocols bgp group MX12-EVPN local-as 100
set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling
set protocols bgp group ToR12-EVPN type external
set protocols bgp group ToR12-EVPN multihop ttl 2
set protocols bgp group ToR12-EVPN multihop no-nexthop-change
set protocols bgp group ToR12-EVPN local-address 192.0.2.11
set protocols bgp group ToR12-EVPN export TEST
set protocols bgp group ToR12-EVPN peer-as 200
set protocols bgp group ToR12-EVPN local-as 100
set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
set protocols l2-learning traceoptions file TOR11-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO term 1 from protocol direct
set policy-options policy-statement LO term 1 from route-filter 192.0.2.11/32
exact
set policy-options policy-statement LO term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.81
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.11:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
TOR11-EVPN-VXLAN-1.log
```

```
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
```

ToR12

```
set system host-name ToR12
set logical-systems CE-1 interfaces ge-1/0/9 unit 0 description "CONNECTED TO
Host 1"
set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-1 interfaces ae1 unit 0 description "CONNECTED TO ToR12"
set logical-systems CE-1 interfaces ae1 unit 0 family bridge interface-mode trunk
set logical-systems CE-1 interfaces ae1 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-1 bridge-domains BD-1 domain-type bridge
set logical-systems CE-1 bridge-domains BD-1 vlan-id 1
set logical-systems CE-1 bridge-domains BD-2 domain-type bridge
set logical-systems CE-1 bridge-domains BD-2 vlan-id 2
set logical-systems CE-1 bridge-domains BD-3 domain-type bridge
set logical-systems CE-1 bridge-domains BD-3 vlan-id 3
set logical-systems CE-1 bridge-domains BD-4 domain-type bridge
set logical-systems CE-1 bridge-domains BD-4 vlan-id 4
set logical-systems CE-1 bridge-domains BD-5 domain-type bridge
set logical-systems CE-1 bridge-domains BD-5 vlan-id 5
set logical-systems CE-3 interfaces ge-1/1/7 unit 0 description "CONNECTED TO
ToR12"
set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-3 interfaces ge-1/1/9 unit 0 description "CONNECTED TO
Host 3"
set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-3 bridge-domains BD-1 domain-type bridge
set logical-systems CE-3 bridge-domains BD-1 vlan-id 1
```

```
set logical-systems CE-3 bridge-domains BD-2 domain-type bridge
set logical-systems CE-3 bridge-domains BD-2 vlan-id 2
set logical-systems CE-3 bridge-domains BD-3 domain-type bridge
set logical-systems CE-3 bridge-domains BD-3 vlan-id 3
set logical-systems CE-3 bridge-domains BD-4 domain-type bridge
set logical-systems CE-3 bridge-domains BD-4 vlan-id 4
set logical-systems CE-3 bridge-domains BD-5 domain-type bridge
set logical-systems CE-3 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 2
set interfaces traceoptions file R1-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces ge-1/0/0 unit 0 description "CONNECTED TO  MX-11"
set interfaces ge-1/0/0 unit 0 family inet address 192.168.6.1/24
set interfaces ge-1/0/4 unit 0 description "CONNECTED TO MX12"
set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.1/24
set interfaces ge-1/0/6 description "CONNECTED TO CE-1"
set interfaces ge-1/0/6 gigether-options 802.3ad ae0
set interfaces ge-1/0/7 unit 0 description "CONNECTED TO CE-3"
set interfaces ge-1/0/7 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/7 unit 0 family bridge vlan-id-list 1-5
set interfaces ge-1/1/0 description "CONNECTED TO ToR11"
set interfaces ge-1/1/0 gigether-options 802.3ad ae1
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR11"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.2/24
set interfaces ge-1/1/6 description "CONNECTED TO ToR12"
set interfaces ge-1/1/6 gigether-options 802.3ad ae1
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options lacp system-id 11:11:11:11:11:11
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces lo0 unit 82 family inet address 192.0.2.12/32
set routing-options router-id 192.0.2.12
set routing-options autonomous-system 200
set routing-options forwarding-table export evpn-pplb
set protocols bgp local-as 200
set protocols bgp group MX11 type external
set protocols bgp group MX11 local-address 192.168.6.1
set protocols bgp group MX11 export LO
set protocols bgp group MX11 export TEST
set protocols bgp group MX11 peer-as 400
set protocols bgp group MX11 local-as 200
set protocols bgp group MX11 neighbor 192.168.6.2 family inet unicast
set protocols bgp group MX12 type external
set protocols bgp group MX12 local-address 192.168.5.1
set protocols bgp group MX12 export LO
set protocols bgp group MX12 export TEST
set protocols bgp group MX12 peer-as 500
set protocols bgp group MX12 local-as 200
set protocols bgp group MX12 neighbor 192.168.5.2 family inet unicast
set protocols bgp group ToR11 type external
set protocols bgp group ToR11 local-address 192.168.2.2
set protocols bgp group ToR11 export LO
set protocols bgp group ToR11 export TEST
set protocols bgp group ToR11 peer-as 100
set protocols bgp group ToR11 local-as 200
```

```
set protocols bgp group ToR11 neighbor 192.168.2.1 family inet unicast
set protocols bgp group MX11-EVPN type external
set protocols bgp group MX11-EVPN multihop ttl 2
set protocols bgp group MX11-EVPN multihop no-nexthop-change
set protocols bgp group MX11-EVPN local-address 192.0.2.12
set protocols bgp group MX11-EVPN export TEST
set protocols bgp group MX11-EVPN peer-as 400
set protocols bgp group MX11-EVPN local-as 200
set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling
set protocols bgp group MX12-EVPN type external
set protocols bgp group MX12-EVPN multihop ttl 2
set protocols bgp group MX12-EVPN multihop no-nexthop-change
set protocols bgp group MX12-EVPN local-address 192.0.2.12
set protocols bgp group MX12-EVPN export TEST
set protocols bgp group MX12-EVPN peer-as 500
set protocols bgp group MX12-EVPN local-as 200
set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling
set protocols bgp group ToR11-EVPN type external
set protocols bgp group ToR11-EVPN multihop ttl 2
set protocols bgp group ToR11-EVPN multihop no-nexthop-change
set protocols bgp group ToR11-EVPN local-address 192.0.2.12
set protocols bgp group ToR11-EVPN export TEST
set protocols bgp group ToR11-EVPN peer-as 100
set protocols bgp group ToR11-EVPN local-as 200
set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
set protocols bgp group ToR12-EVPN export TEST
set protocols l2-learning traceoptions file TOR12-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO term 1 from protocol direct
set policy-options policy-statement LO term 1 from route-filter 192.0.2.12/32
exact
set policy-options policy-statement LO term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.82
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/7.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.12:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
TOR12-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
```

```
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
```

Data Center Gateway and WAN Edge 1 Router (MX11)

```
set system host-name MX11
set interfaces ge-5/1/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-5/1/0 unit 0 family inet address 192.168.7.1/24
set interfaces lt-5/1/0 esi 00:22:22:22:22:22:22:22:22:22
set interfaces lt-5/1/0 esi all-active
set interfaces lt-5/1/0 unit 0 peer-unit 1
set interfaces lt-5/1/0 unit 0 family bridge interface-mode trunk
set interfaces lt-5/1/0 unit 0 family bridge vlan-id-list 1-5
set interfaces lt-5/1/0 unit 1 peer-unit 0
set interfaces lt-5/1/0 unit 1 family bridge interface-mode trunk
set interfaces lt-5/1/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-5/1/1 unit 0 description "CONNECTED TO ToR11"
set interfaces ge-5/1/1 unit 0 family inet address 192.168.3.2/24
set interfaces ge-5/1/8 unit 0 description "CONNECTED TO ToR12"
set interfaces ge-5/1/8 unit 0 family inet address 192.168.6.2/24
set interfaces ge-5/1/9 unit 0 description "CONNECTED TO P"
set interfaces ge-5/1/9 unit 0 family inet address 203.0.1.1/24
set interfaces ge-5/1/9 unit 0 family mpls
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:11:aa:aa:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
set interfaces irb unit 1 family inet address 10.11.1.12/24 virtual-gateway-address
 10.11.1.10
set interfaces irb unit 2 proxy-macip-advertisement
set interfaces irb unit 2 virtual-gateway-esi 00:11:bb:bb:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.12/24 virtual-gateway-address
 10.12.1.10
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:11:cc:cc:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.12/24 virtual-gateway-address
 10.13.1.10
set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:11:dd:dd:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.12/24 virtual-gateway-address
 10.14.1.10
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:11:ee:ee:ee:ee:ee:ee:ee:ee
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.12/24 virtual-gateway-address
 10.15.1.10
set interfaces lo0 unit 84 family inet address 192.0.2.21/32
set interfaces lo0 unit 84 family mpls
```

```
set routing-options router-id 192.0.2.21
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX11-TO-MX12 to 192.0.2.22
set protocols mpls label-switched-path MX11-TO-P to 203.0.113.1
set protocols mpls label-switched-path MX11-TO-MX21 to 198.51.100.21
set protocols mpls label-switched-path MX11-TO-MX22 to 198.51.100.22
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 192.0.2.21
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT local-address 192.0.2.21
set protocols bgp group INT family evpn signaling
set protocols bgp group INT export TEST
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR11 type external
set protocols bgp group ToR11 local-address 192.168.3.2
set protocols bgp group ToR11 import TEST
set protocols bgp group ToR11 export TEST
set protocols bgp group ToR11 export LO
set protocols bgp group ToR11 peer-as 100
set protocols bgp group ToR11 local-as 400
set protocols bgp group ToR11 neighbor 192.168.3.1 family inet unicast
set protocols bgp group ToR12 type external
set protocols bgp group ToR12 local-address 192.168.6.2
set protocols bgp group ToR12 export TEST
set protocols bgp group ToR12 export LO
set protocols bgp group ToR12 peer-as 200
set protocols bgp group ToR12 local-as 400
set protocols bgp group ToR12 neighbor 192.168.6.1 family inet unicast
set protocols bgp group MX12 type external
set protocols bgp group MX12 local-address 192.168.7.1
set protocols bgp group MX12 export TEST
set protocols bgp group MX12 export LO
set protocols bgp group MX12 peer-as 500
set protocols bgp group MX12 local-as 400
set protocols bgp group MX12 neighbor 192.168.7.2 family inet unicast
set protocols bgp group ToR11-EVPN type external
set protocols bgp group ToR11-EVPN multihop ttl 2
set protocols bgp group ToR11-EVPN multihop no-nexthop-change
set protocols bgp group ToR11-EVPN local-address 192.0.2.21
set protocols bgp group ToR11-EVPN export TEST
set protocols bgp group ToR11-EVPN peer-as 100
set protocols bgp group ToR11-EVPN local-as 400
set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
set protocols bgp group ToR12-EVPN type external
set protocols bgp group ToR12-EVPN multihop ttl 2
set protocols bgp group ToR12-EVPN multihop no-nexthop-change
set protocols bgp group ToR12-EVPN local-address 192.0.2.21
set protocols bgp group ToR12-EVPN export TEST
set protocols bgp group ToR12-EVPN peer-as 200
set protocols bgp group ToR12-EVPN local-as 400
set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
set protocols bgp group MX12-EVPN type external
set protocols bgp group MX12-EVPN multihop ttl 2
set protocols bgp group MX12-EVPN multihop no-nexthop-change
```

```
set protocols bgp group MX12-EVPN local-address 192.0.2.21
set protocols bgp group MX12-EVPN export TEST
set protocols bgp group MX12-EVPN peer-as 500
set protocols bgp group MX12-EVPN local-as 400
set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn signaling
set protocols bgp group MX11-EVPN export TEST
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-5/1/9.0
set protocols ospf area 0.0.0.0 interface lo0.84 passive
set protocols l2-learning traceoptions file MX11-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO term 1 from protocol direct
set policy-options policy-statement LO term 1 from route-filter 192.0.2.21/32
exact
set policy-options policy-statement LO term 1 then accept
set policy-options policy-statement LO from protocol direct
set policy-options policy-statement LO from route-filter 192.0.2.21/32 exact
set policy-options policy-statement LO then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
set routing-instances EVPN-MPLS-1 interface lt-5/1/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 192.0.2.21:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file
MX11-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.84
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-5/1/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.21:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
MX11-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
```

```
no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10
```

Data Center Gateway and WAN Edge 2 Router (MX12)

```
set system host-name MX12
set logical-systems P interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX21"
set logical-systems P interfaces ge-1/0/0 unit 0 family inet address 203.0.4.2/24
set logical-systems P interfaces ge-1/0/0 unit 0 family mpls
set logical-systems P interfaces ge-1/0/2 unit 0 description "CONNECTED TO MX22"
set logical-systems P interfaces ge-1/0/2 unit 0 family inet address 203.0.3.2/24
set logical-systems P interfaces ge-1/0/2 unit 0 family mpls
set logical-systems P interfaces ge-1/1/1 unit 0 description "CONNECTED TO MX11"
set logical-systems P interfaces ge-1/1/1 unit 0 family inet address 203.0.1.2/24
set logical-systems P interfaces ge-1/1/1 unit 0 family mpls
set logical-systems P interfaces ge-1/1/6 unit 0 description "CONNECTED TO MX12"
set logical-systems P interfaces ge-1/1/6 unit 0 family inet address 203.0.2.2/24
set logical-systems P interfaces ge-1/1/6 unit 0 family mpls
set logical-systems P interfaces lo0 unit 86 family inet address 203.0.113.1/32
set logical-systems P interfaces lo0 unit 86 family mpls
set logical-systems P protocols rsvp interface all
set logical-systems P protocols mpls label-switched-path P-TO-MX11 from 203.0.113.1
set logical-systems P protocols mpls label-switched-path P-TO-MX11 to 192.0.2.21
set logical-systems P protocols mpls label-switched-path P-TO-MX12 to 192.0.2.22
set logical-systems P protocols mpls label-switched-path P-TO-MX21 to 198.51.100.21
set logical-systems P protocols mpls label-switched-path P-TO-MX22 to 198.51.100.22
set logical-systems P protocols mpls interface all
set logical-systems P protocols bgp local-address 203.0.113.1
set logical-systems P protocols bgp local-as 300
```

```
set logical-systems P protocols bgp group INT type internal
set logical-systems P protocols bgp group INT import BLOCK-VXLAN-ROUTES-FROM-CORE
set logical-systems P protocols bgp group INT family evpn signaling
set logical-systems P protocols bgp group INT cluster 203.0.113.1
set logical-systems P protocols bgp group INT neighbor 192.0.2.21
set logical-systems P protocols bgp group INT neighbor 192.0.2.22
set logical-systems P protocols bgp group INT neighbor 198.51.100.21
set logical-systems P protocols bgp group INT neighbor 198.51.100.22
set logical-systems P protocols ospf traffic-engineering
set logical-systems P protocols ospf area 0.0.0.0 interface all
set logical-systems P protocols ospf area 0.0.0.0 interface lo0.86
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 1 from protocol bgp
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 1 from community RT-CORE
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 1 then accept
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 2 from protocol bgp
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 2 from community RT-DC1
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 2 then reject
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 3 from protocol bgp
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 3 from community RT-DC2
set logical-systems P policy-options policy-statement BLOCK-VXLAN-ROUTES-FROM-CORE
 term 3 then reject
set logical-systems P policy-options community RT-CORE members target:1:2
set logical-systems P policy-options community RT-DC1 members target:1:1
set logical-systems P policy-options community RT-DC2 members target:1:3
set logical-systems P routing-options router-id 203.0.113.1
set logical-systems P routing-options autonomous-system 300
set chassis fpc 1 pic 0 tunnel-services
set chassis network-services enhanced-ip
set interfaces traceoptions file R3-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces lt-1/0/0 esi 00:22:22:22:22:22:22:22:22:22
set interfaces lt-1/0/0 esi all-active
set interfaces lt-1/0/0 unit 0 peer-unit 1
set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5
set interfaces lt-1/0/0 unit 1 peer-unit 0
set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-1/0/4 unit 0 description "CONNECTED TO ToR12"
set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.2/24
set interfaces ge-1/0/5 unit 0 description "CONNECTED TO TOR11"
set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.2/24
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO P"
set interfaces ge-1/0/6 unit 0 family inet address 203.0.2.1/24
set interfaces ge-1/0/6 unit 0 family mpls
set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX11"
set interfaces ge-1/1/0 unit 0 family inet address 192.168.7.2/24
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:11:aa:aa:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
```

```
set interfaces irb unit 1 family inet address 10.11.1.13/24 virtual-gateway-address
 10.11.1.10
set interfaces irb unit 2 proxy-macip-advertisement
set interfaces irb unit 2 virtual-gateway-esi 00:11:bb:bb:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.13/24 virtual-gateway-address
 10.12.1.10
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:11:cc:cc:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.13/24 virtual-gateway-address
 10.13.1.10
set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:11:dd:dd:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.13/24 virtual-gateway-address
 10.14.1.10
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:11:ee:ee:ee:ee:ee:ee:ee:ee
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.13/24 virtual-gateway-address
 10.15.1.10
set interfaces lo0 unit 85 family inet address 192.0.2.22/32
set interfaces lo0 unit 85 family mpls
set routing-options router-id 192.0.2.22
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX12-TO-MX11 to 192.0.2.21
set protocols mpls label-switched-path MX12-TO-P to 203.0.113.1
set protocols mpls label-switched-path MX12-TO-MX21 to 198.51.100.21
set protocols mpls label-switched-path MX12-TO-MX22 to 198.51.100.22
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 192.0.2.22
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT family evpn signaling
set protocols bgp group INT export TEST
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR11 type external
set protocols bgp group ToR11 local-address 192.168.4.2
set protocols bgp group ToR11 export TEST
set protocols bgp group ToR11 export LO
set protocols bgp group ToR11 peer-as 100
set protocols bgp group ToR11 local-as 500
set protocols bgp group ToR11 neighbor 192.168.4.1 family inet unicast
set protocols bgp group ToR12 type external
set protocols bgp group ToR12 local-address 192.168.5.2
set protocols bgp group ToR12 export TEST
set protocols bgp group ToR12 export LO
set protocols bgp group ToR12 peer-as 200
set protocols bgp group ToR12 local-as 500
set protocols bgp group ToR12 neighbor 192.168.5.1 family inet unicast
set protocols bgp group MX11 type external
set protocols bgp group MX11 local-address 192.168.7.2
set protocols bgp group MX11 export TEST
set protocols bgp group MX11 export LO
```

```
set protocols bgp group MX11 peer-as 400
set protocols bgp group MX11 local-as 500
set protocols bgp group MX11 neighbor 192.168.7.1 family inet unicast
set protocols bgp group ToR11-EVPN type external
set protocols bgp group ToR11-EVPN multihop ttl 2
set protocols bgp group ToR11-EVPN multihop no-nexthop-change
set protocols bgp group ToR11-EVPN local-address 192.0.2.22
set protocols bgp group ToR11-EVPN export TEST
set protocols bgp group ToR11-EVPN peer-as 100
set protocols bgp group ToR11-EVPN local-as 500
set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family evpn signaling
set protocols bgp group ToR12-EVPN type external
set protocols bgp group ToR12-EVPN multihop ttl 2
set protocols bgp group ToR12-EVPN multihop no-nexthop-change
set protocols bgp group ToR12-EVPN local-address 192.0.2.22
set protocols bgp group ToR12-EVPN export TEST
set protocols bgp group ToR12-EVPN peer-as 200
set protocols bgp group ToR12-EVPN local-as 500
set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family evpn signaling
set protocols bgp group MX11-EVPN type external
set protocols bgp group MX11-EVPN multihop ttl 2
set protocols bgp group MX11-EVPN multihop no-nexthop-change
set protocols bgp group MX11-EVPN local-address 192.0.2.22
set protocols bgp group MX11-EVPN export TEST
set protocols bgp group MX11-EVPN peer-as 400
set protocols bgp group MX11-EVPN local-as 500
set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn signaling
set protocols bgp group MX12-EVPN export TEST
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/0/6.0
set protocols ospf area 0.0.0.0 interface lo0.85 passive
set protocols l2-learning traceoptions file MX12-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO term 1 from protocol direct
set policy-options policy-statement LO term 1 from route-filter 192.0.2.22/32
exact
set policy-options policy-statement LO term 1 then accept
set policy-options policy-statement LO from protocol direct
set policy-options policy-statement LO from route-filter 192.0.2.22/32 exact
set policy-options policy-statement LO then accept
set policy-options policy-statement TEST from protocol bgp
set policy-options policy-statement TEST from protocol evpn
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 192.0.2.22:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file
MX12-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
```

```
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.85
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.22:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
MX12-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-4
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10
```

Data Center Gateway and WAN Edge 3 Router (MX21)

```
set system host-name MX21
set interfaces ge-3/0/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-3/0/0 unit 0 family inet address 192.168.13.1/24
set interfaces ge-3/1/0 unit 0 description "CONNECTED TO ToR22"
set interfaces ge-3/1/0 unit 0 family inet address 192.168.8.1/24
set interfaces ge-5/0/0 unit 0 description "CONNECTED TO P"
set interfaces ge-5/0/0 unit 0 family inet address 203.0.4.1/24
set interfaces ge-5/0/0 unit 0 family mpls
set interfaces lt-5/0/0 esi 00:33:33:33:33:33:33:33:33:33
set interfaces lt-5/0/0 esi all-active
set interfaces lt-5/0/0 unit 0 peer-unit 1
set interfaces lt-5/0/0 unit 0 family bridge interface-mode trunk
set interfaces lt-5/0/0 unit 0 family bridge vlan-id-list 1-5
set interfaces lt-5/0/0 unit 1 peer-unit 0
set interfaces lt-5/0/0 unit 1 family bridge interface-mode trunk
set interfaces lt-5/0/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-5/0/1 unit 0 description "CONNECTED TO ToR21"
set interfaces ge-5/0/1 unit 0 family inet address 192.168.9.1/24
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:22:aa:aa:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
set interfaces irb unit 1 family inet address 10.11.1.14/24 virtual-gateway-address
 10.11.1.11
set interfaces irb unit 2 proxy-macip-advertisement
set interfaces irb unit 2 virtual-gateway-esi 00:22:bb:bb:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.14/24 virtual-gateway-address
 10.12.1.11
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:22:cc:cc:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.14/24 virtual-gateway-address
 10.13.1.11
set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:22:dd:dd:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.14/24 virtual-gateway-address
 10.14.1.11
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:22:ee:ee:ee:ee:ee:ee:ee:ee
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.14/24 virtual-gateway-address
 10.15.1.11
set interfaces lo0 unit 87 family inet address 198.51.100.21/32
set interfaces lo0 unit 87 family mpls
set routing-options router-id 198.51.100.21
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX21-TO-MX11 to 192.0.2.21
set protocols mpls label-switched-path MX21-TO-MX12 to 192.0.2.22
set protocols mpls label-switched-path MX21-TO-P to 203.0.113.1
set protocols mpls label-switched-path MX21-TO-MX22 to 198.51.100.22
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
```

```
set protocols bgp local-address 198.51.100.21
set protocols bgp export TEST
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT local-address 198.51.100.21
set protocols bgp group INT family evpn signaling
set protocols bgp group INT export TEST1
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR21 type external
set protocols bgp group ToR21 local-address 192.168.9.1
set protocols bgp group ToR21 export TEST
set protocols bgp group ToR21 export LO
set protocols bgp group ToR21 peer-as 600
set protocols bgp group ToR21 local-as 800
set protocols bgp group ToR21 neighbor 192.168.9.2 family inet unicast
set protocols bgp group ToR22 type external
set protocols bgp group ToR22 local-address 192.168.8.1
set protocols bgp group ToR22 export TEST
set protocols bgp group ToR22 export LO
set protocols bgp group ToR22 peer-as 700
set protocols bgp group ToR22 local-as 800
set protocols bgp group ToR22 neighbor 192.168.8.2 family inet unicast
set protocols bgp group MX22 type external
set protocols bgp group MX22 local-address 192.168.13.1
set protocols bgp group MX22 export TEST
set protocols bgp group MX22 export LO
set protocols bgp group MX22 peer-as 900
set protocols bgp group MX22 local-as 800
set protocols bgp group MX22 neighbor 10.115.15.2 family inet unicast
set protocols bgp group ToR21-EVPN type external
set protocols bgp group ToR21-EVPN multihop ttl 2
set protocols bgp group ToR21-EVPN multihop no-nexthop-change
set protocols bgp group ToR21-EVPN local-address 198.51.100.21
set protocols bgp group ToR21-EVPN peer-as 600
set protocols bgp group ToR21-EVPN local-as 800
set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
set protocols bgp group ToR22-EVPN type external
set protocols bgp group ToR22-EVPN multihop ttl 2
set protocols bgp group ToR22-EVPN multihop no-nexthop-change
set protocols bgp group ToR22-EVPN local-address 198.51.100.21
set protocols bgp group ToR22-EVPN peer-as 700
set protocols bgp group ToR22-EVPN local-as 800
set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
set protocols bgp group MX22-EVPN type external
set protocols bgp group MX22-EVPN multihop ttl 2
set protocols bgp group MX22-EVPN multihop no-nexthop-change
set protocols bgp group MX22-EVPN local-address 198.51.100.21
set protocols bgp group MX22-EVPN peer-as 900
set protocols bgp group MX22-EVPN local-as 800
set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn signaling
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-5/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.87 passive
set protocols l2-learning traceoptions file MX21-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO from protocol direct
set policy-options policy-statement LO from route-filter 198.51.100.21/32 exact
```

```
set policy-options policy-statement LO then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement TEST1 term 1 from protocol bgp
set policy-options policy-statement TEST1 term 1 from external
set policy-options policy-statement TEST1 term 1 then reject
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
set routing-instances EVPN-MPLS-1 interface lt-5/0/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 198.51.100.21:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file
MX21-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.87
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-5/0/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.21:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
MX21-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
```

```
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10
```

Data Center Gateway and WAN Edge 4 Router (MX22)

```
set system host-name MX22
set interfaces xe-0/0/0 unit 0 description "CONNECTED TO ToR22"
set interfaces xe-0/0/0 unit 0 family inet address 192.168.11.1/24
set interfaces xe-0/0/1 unit 0 description "CONNECTED TO ToR21"
set interfaces xe-0/0/1 unit 0 family inet address 192.168.10.1/24
set interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-1/0/0 unit 0 family inet address 10.115.15.2/24
set interfaces lt-1/0/0 esi 00:33:33:33:33:33:33:33:33:33
set interfaces lt-1/0/0 esi all-active
set interfaces lt-1/0/0 unit 0 peer-unit 1
set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5
set interfaces lt-1/0/0 unit 1 peer-unit 0
set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5
set interfaces ge-1/0/2 unit 0 description "CONNECTED TO P"
set interfaces ge-1/0/2 unit 0 family inet address 203.0.3.1/24
set interfaces ge-1/0/2 unit 0 family mpls
set interfaces irb unit 1 proxy-macip-advertisement
set interfaces irb unit 1 virtual-gateway-esi 00:22:aa:aa:aa:aa:aa:aa:aa:aa
set interfaces irb unit 1 virtual-gateway-esi all-active
set interfaces irb unit 1 family inet address 10.11.1.15/24 virtual-gateway-address
 10.11.1.11
set interfaces irb unit 2 proxy-macip-advertisement
set interfaces irb unit 2 virtual-gateway-esi 00:22:bb:bb:bb:bb:bb:bb:bb:bb
set interfaces irb unit 2 virtual-gateway-esi all-active
set interfaces irb unit 2 family inet address 10.12.1.15/24 virtual-gateway-address
 10.12.1.11
set interfaces irb unit 3 proxy-macip-advertisement
set interfaces irb unit 3 virtual-gateway-esi 00:22:cc:cc:cc:cc:cc:cc:cc:cc
set interfaces irb unit 3 virtual-gateway-esi all-active
set interfaces irb unit 3 family inet address 10.13.1.15/24 virtual-gateway-address
 10.13.1.11
set interfaces irb unit 4 proxy-macip-advertisement
set interfaces irb unit 4 virtual-gateway-esi 00:22:dd:dd:dd:dd:dd:dd:dd:dd
set interfaces irb unit 4 virtual-gateway-esi all-active
set interfaces irb unit 4 family inet address 10.14.1.15/24 virtual-gateway-address
 10.14.1.11
set interfaces irb unit 5 proxy-macip-advertisement
set interfaces irb unit 5 virtual-gateway-esi 00:22:ee:ee:ee:ee:ee:ee:ee:ee
```

```
set interfaces irb unit 5 virtual-gateway-esi all-active
set interfaces irb unit 5 family inet address 10.15.1.15/24 virtual-gateway-address
 10.15.1.11
set interfaces lo0 unit 88 family inet address 198.51.100.22/32
set routing-options router-id 198.51.100.22
set routing-options autonomous-system 300
set routing-options forwarding-table export evpn-pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path MX22-TO-MX11 to 192.0.2.21
set protocols mpls label-switched-path MX22-TO-MX12 to 192.0.2.22
set protocols mpls label-switched-path MX22-TO-P to 203.0.113.1
set protocols mpls label-switched-path MX22-TO-MX21 to 198.51.100.21
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 198.51.100.22
set protocols bgp export TEST
set protocols bgp local-as 300
set protocols bgp group INT type internal
set protocols bgp group INT family evpn signaling
set protocols bgp group INT export TEST1
set protocols bgp group INT neighbor 203.0.113.1
set protocols bgp group ToR21 type external
set protocols bgp group ToR21 local-address 192.168.10.1
set protocols bgp group ToR21 export TEST
set protocols bgp group ToR21 export LO
set protocols bgp group ToR21 peer-as 600
set protocols bgp group ToR21 local-as 900
set protocols bgp group ToR21 neighbor 10.102.2.1 family inet unicast
set protocols bgp group ToR22 type external
set protocols bgp group ToR22 local-address 192.168.11.1
set protocols bgp group ToR22 export TEST
set protocols bgp group ToR22 export LO
set protocols bgp group ToR22 peer-as 700
set protocols bgp group ToR22 local-as 900
set protocols bgp group ToR22 neighbor 192.168.11.2 family inet unicast
set protocols bgp group MX21 type external
set protocols bgp group MX21 local-address 10.115.15.2
set protocols bgp group MX21 export TEST
set protocols bgp group MX21 export LO
set protocols bgp group MX21 peer-as 800
set protocols bgp group MX21 local-as 900
set protocols bgp group MX21 neighbor 192.168.13.1 family inet unicast
set protocols bgp group ToR21-EVPN type external
set protocols bgp group ToR21-EVPN multihop ttl 2
set protocols bgp group ToR21-EVPN multihop no-nexthop-change
set protocols bgp group ToR21-EVPN local-address 198.51.100.22
set protocols bgp group ToR21-EVPN peer-as 600
set protocols bgp group ToR21-EVPN local-as 900
set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
set protocols bgp group ToR22-EVPN type external
set protocols bgp group ToR22-EVPN multihop ttl 2
set protocols bgp group ToR22-EVPN multihop no-nexthop-change
set protocols bgp group ToR22-EVPN local-address 198.51.100.22
set protocols bgp group ToR22-EVPN peer-as 700
set protocols bgp group ToR22-EVPN local-as 900
set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
set protocols bgp group MX21-EVPN type external
set protocols bgp group MX21-EVPN multihop ttl 2
```

```
set protocols bgp group MX21-EVPN multihop no-nexthop-change
set protocols bgp group MX21-EVPN local-address 198.51.100.22
set protocols bgp group MX21-EVPN peer-as 800
set protocols bgp group MX21-EVPN local-as 900
set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn signaling
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/0/2.0
set protocols ospf area 0.0.0.0 interface lo0.88 passive
set protocols l2-learning traceoptions file MX22-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO from protocol direct
set policy-options policy-statement LO from route-filter 198.51.100.22/32 exact
set policy-options policy-statement LO then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement TEST1 term 1 from protocol bgp
set policy-options policy-statement TEST1 term 1 from external
set policy-options policy-statement TEST1 term 1 then reject
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-MPLS-1 instance-type virtual-switch
set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
set routing-instances EVPN-MPLS-1 route-distinguisher 198.51.100.22:100
set routing-instances EVPN-MPLS-1 vrf-target target:1:2
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file
MX22-EVPN-MPLS-1.log
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-MPLS-1 protocols evpn traceoptions flag all
set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list 1-5
set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
no-gateway-community
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.88
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.22:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
MX22-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
no-gateway-community
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
```

```
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 routing-interface irb.1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 routing-interface irb.2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 routing-interface irb.3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 routing-interface irb.4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 routing-interface irb.5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
set routing-instances VRF instance-type vrf
set routing-instances VRF interface irb.1
set routing-instances VRF interface irb.2
set routing-instances VRF interface irb.3
set routing-instances VRF interface irb.4
set routing-instances VRF interface irb.5
set routing-instances VRF route-distinguisher 1:1
set routing-instances VRF vrf-target target:10:10
```

ToR–21

```
set system host-name ToR21
set logical-systems CE-4 interfaces ge-1/0/9 unit 0 description "CONNECTED TO
Host 4"
set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-4 interfaces ge-1/1/6 unit 0 description "CONNECTED TO
ToR21"
set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-4 bridge-domains BD-1 domain-type bridge
set logical-systems CE-4 bridge-domains BD-1 vlan-id 1
set logical-systems CE-4 bridge-domains BD-2 domain-type bridge
set logical-systems CE-4 bridge-domains BD-2 vlan-id 2
set logical-systems CE-4 bridge-domains BD-3 domain-type bridge
set logical-systems CE-4 bridge-domains BD-3 vlan-id 3
set logical-systems CE-4 bridge-domains BD-4 domain-type bridge
set logical-systems CE-4 bridge-domains BD-4 vlan-id 4
set logical-systems CE-4 bridge-domains BD-5 domain-type bridge
set logical-systems CE-4 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 1
set interfaces traceoptions file R6-DCD.log
set interfaces traceoptions file size 10m
```

```
set interfaces traceoptions flag all
set interfaces xe-0/0/0 unit 0 description "CONNECTED TO MX22"
set interfaces xe-0/0/0 unit 0 family inet address 192.168.10.2/24
set interfaces ge-1/0/0 description "CONNECTED TO CE-5"
set interfaces ge-1/0/0 gigether-options 802.3ad ae0
set interfaces ge-1/0/1 unit 0 description "CONNECTED TO MX21"
set interfaces ge-1/0/1 unit 0 family inet address 192.168.101.1/24
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-4"
set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR22"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.12.1/24
set interfaces ae0 esi 00:44:44:44:44:44:44:44:44:44
set interfaces ae0 esi all-active
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 22:22:22:22:22:22
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces lo0 unit 90 family inet address 198.51.100.11/32
set routing-options router-id 198.51.100.11
set routing-options autonomous-system 600
set routing-options forwarding-table export evpn-pplb
set protocols bgp export TEST
set protocols bgp local-as 600
set protocols bgp group MX21 type external
set protocols bgp group MX21 local-address 192.168.9.2
set protocols bgp group MX21 export LO
set protocols bgp group MX21 export TEST
set protocols bgp group MX21 peer-as 800
set protocols bgp group MX21 local-as 600
set protocols bgp group MX21 neighbor 192.168.9.1 family inet unicast
set protocols bgp group MX22 type external
set protocols bgp group MX22 local-address 10.102.2.1
set protocols bgp group MX22 export LO
set protocols bgp group MX22 export TEST
set protocols bgp group MX22 peer-as 900
set protocols bgp group MX22 local-as 600
set protocols bgp group MX22 neighbor 192.168.10.1 family inet unicast
set protocols bgp group ToR22 type external
set protocols bgp group ToR22 local-address 10.105.5.1
set protocols bgp group ToR22 export LO
set protocols bgp group ToR22 export TEST
set protocols bgp group ToR22 peer-as 700
set protocols bgp group ToR22 local-as 600
set protocols bgp group ToR22 neighbor 192.168.12.2 family inet unicast
set protocols bgp group MX21-EVPN type external
set protocols bgp group MX21-EVPN multihop ttl 2
set protocols bgp group MX21-EVPN multihop no-nexthop-change
set protocols bgp group MX21-EVPN local-address 198.51.100.11
set protocols bgp group MX21-EVPN peer-as 800
set protocols bgp group MX21-EVPN local-as 600
set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn signaling
set protocols bgp group MX22-EVPN type external
set protocols bgp group MX22-EVPN multihop ttl 2
set protocols bgp group MX22-EVPN multihop no-nexthop-change
set protocols bgp group MX22-EVPN local-address 198.51.100.11
set protocols bgp group MX22-EVPN peer-as 900
set protocols bgp group MX22-EVPN local-as 600
```

```
set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn signaling
set protocols bgp group ToR22-EVPN type external
set protocols bgp group ToR22-EVPN multihop ttl 2
set protocols bgp group ToR22-EVPN multihop no-nexthop-change
set protocols bgp group ToR22-EVPN local-address 198.51.100.11
set protocols bgp group ToR22-EVPN peer-as 700
set protocols bgp group ToR22-EVPN local-as 600
set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family evpn signaling
set protocols l2-learning traceoptions file TOR21-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO term 1 from protocol direct
set policy-options policy-statement LO term 1 from route-filter 198.51.100.11/32
 exact
set policy-options policy-statement LO term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.90
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.11:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
TOR21-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
```

ToR–22

```
set system host-name ToR22
set logical-systems CE-5 interfaces ge-1/0/9 unit 0 description "CONNECTED TO
Host 5"
set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge interface-mode
```

```
 trunk
set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-5 interfaces ae1 unit 0 description "CONNECTED TO ToR21"
set logical-systems CE-5 interfaces ae1 unit 0 family bridge interface-mode trunk
set logical-systems CE-5 interfaces ae1 unit 0 family bridge vlan-id-list 1-5
set logical-systems CE-5 bridge-domains BD-1 domain-type bridge
set logical-systems CE-5 bridge-domains BD-1 vlan-id 1
set logical-systems CE-5 bridge-domains BD-2 domain-type bridge
set logical-systems CE-5 bridge-domains BD-2 vlan-id 2
set logical-systems CE-5 bridge-domains BD-3 domain-type bridge
set logical-systems CE-5 bridge-domains BD-3 vlan-id 3
set logical-systems CE-5 bridge-domains BD-4 domain-type bridge
set logical-systems CE-5 bridge-domains BD-4 vlan-id 4
set logical-systems CE-5 bridge-domains BD-5 domain-type bridge
set logical-systems CE-5 bridge-domains BD-5 vlan-id 5
set logical-systems CE-6 interfaces ge-1/1/6 unit 0 description "CONNECTED TO
ToR22"
set logical-systems CE-6 interfaces ge-1/1/6 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-6 interfaces ge-1/1/6 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-6 interfaces ge-1/1/9 unit 0 description "CONNECTED TO
Host 6"
set logical-systems CE-6 interfaces ge-1/1/9 unit 0 family bridge interface-mode
 trunk
set logical-systems CE-6 interfaces ge-1/1/9 unit 0 family bridge vlan-id-list
1-5
set logical-systems CE-6 bridge-domains BD-1 domain-type bridge
set logical-systems CE-6 bridge-domains BD-1 vlan-id 1
set logical-systems CE-6 bridge-domains BD-2 domain-type bridge
set logical-systems CE-6 bridge-domains BD-2 vlan-id 2
set logical-systems CE-6 bridge-domains BD-3 domain-type bridge
set logical-systems CE-6 bridge-domains BD-3 vlan-id 3
set logical-systems CE-6 bridge-domains BD-4 domain-type bridge
set logical-systems CE-6 bridge-domains BD-4 vlan-id 4
set logical-systems CE-6 bridge-domains BD-5 domain-type bridge
set logical-systems CE-6 bridge-domains BD-5 vlan-id 5
set chassis aggregated-devices ethernet device-count 2
set interfaces traceoptions file R7-DCD.log
set interfaces traceoptions file size 10m
set interfaces traceoptions flag all
set interfaces xe-0/0/0 unit 0 description "CONNECTED TO MX22"
set interfaces xe-0/0/0 unit 0 family inet address 192.168.11.2/24
set interfaces ge-1/0/0 description "CONNECTED TO ToR21"
set interfaces ge-1/0/0 gigether-options 802.3ad ae1
set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-6"
set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
set interfaces ge-1/0/7 description "CONNECTED TO ToR22"
set interfaces ge-1/0/7 gigether-options 802.3ad ae1
set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX21"
set interfaces ge-1/1/0 unit 0 family inet address 192.168.8.2/24
set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR21"
set interfaces ge-1/1/3 unit 0 family inet address 192.168.12.2/24
set interfaces ge-1/1/7 description "CONNECTED TO CE-5"
set interfaces ge-1/1/7 gigether-options 802.3ad ae0
set interfaces ae0 esi 00:44:44:44:44:44:44:44:44:44
set interfaces ae0 esi all-active
```

```
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 22:22:22:22:22:22
set interfaces ae0 unit 0 family bridge interface-mode trunk
set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 aggregated-ether-options lacp periodic fast
set interfaces ae1 aggregated-ether-options lacp system-id 22:22:22:22:22:22
set interfaces lo0 unit 92 family inet address 198.51.100.12/32
set routing-options router-id 198.51.100.12
set routing-options autonomous-system 700
set routing-options forwarding-table export evpn-pplb
set protocols bgp export TEST
set protocols bgp local-as 700
set protocols bgp group MX21 type external
set protocols bgp group MX21 local-address 192.168.8.2
set protocols bgp group MX21 export LO
set protocols bgp group MX21 export TEST
set protocols bgp group MX21 peer-as 800
set protocols bgp group MX21 local-as 700
set protocols bgp group MX21 neighbor 192.168.8.1 family inet unicast
set protocols bgp group MX22 type external
set protocols bgp group MX22 local-address 192.168.11.2
set protocols bgp group MX22 export LO
set protocols bgp group MX22 export TEST
set protocols bgp group MX22 peer-as 900
set protocols bgp group MX22 local-as 700
set protocols bgp group MX22 neighbor 192.168.11.1 family inet unicast
set protocols bgp group ToR21 type external
set protocols bgp group ToR21 local-address 192.168.12.2
set protocols bgp group ToR21 export LO
set protocols bgp group ToR21 export TEST
set protocols bgp group ToR21 peer-as 600
set protocols bgp group ToR21 local-as 700
set protocols bgp group ToR21 neighbor 10.105.5.1 family inet unicast
set protocols bgp group MX21-EVPN type external
set protocols bgp group MX21-EVPN multihop ttl 2
set protocols bgp group MX21-EVPN multihop no-nexthop-change
set protocols bgp group MX21-EVPN local-address 198.51.100.12
set protocols bgp group MX21-EVPN peer-as 800
set protocols bgp group MX21-EVPN local-as 700
set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family evpn signaling
set protocols bgp group MX22-EVPN type external
set protocols bgp group MX22-EVPN multihop ttl 2
set protocols bgp group MX22-EVPN multihop no-nexthop-change
set protocols bgp group MX22-EVPN local-address 198.51.100.12
set protocols bgp group MX22-EVPN peer-as 900
set protocols bgp group MX22-EVPN local-as 700
set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family evpn signaling
set protocols bgp group ToR21-EVPN type external
set protocols bgp group ToR21-EVPN multihop ttl 2
set protocols bgp group ToR21-EVPN multihop no-nexthop-change
set protocols bgp group ToR21-EVPN local-address 198.51.100.12
set protocols bgp group ToR21-EVPN peer-as 600
set protocols bgp group ToR21-EVPN local-as 700
set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family evpn signaling
set protocols l2-learning traceoptions file TOR22-L2ALD.log
set protocols l2-learning traceoptions file size 10m
set protocols l2-learning traceoptions level all
```

```
set protocols l2-learning traceoptions flag all
set policy-options policy-statement LO term 1 from protocol direct
set policy-options policy-statement LO term 1 from route-filter 198.51.100.12/32
 exact
set policy-options policy-statement LO term 1 then accept
set policy-options policy-statement TEST then community add NO-EXPORT
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set policy-options community NO-EXPORT members no-advertise
set policy-options community NO-EXPORT members no-export
set policy-options community NO-EXPORT members no-export-subconfed
set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.92
set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
set routing-instances EVPN-VXLAN-1 interface ae0.0
set routing-instances EVPN-VXLAN-1 route-distinguisher 198.51.100.12:1
set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file
TOR22-EVPN-VXLAN-1.log
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions file size 10m
set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions flag all
set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation vxlan
set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list 1-5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni 1
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni 2
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni 3
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni 4
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type bridge
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni 5
```

Configuring ToR11

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router as ToR11:

1. Set the system hostname.

```
[edit]
```

```
user@ToR11# set system host-name ToR11
```

2. Configure the interfaces and bridge domains on the CE2 device to enable Layer 2 connectivity.

```
[edit]
```

```
user@ce2# set logical-systems CE-2 interfaces ge-1/0/9 unit 0 description
"CONNECTED TO Host 2"
user@ce2# set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge
 interface-mode trunk
user@ce2# set logical-systems CE-2 interfaces ge-1/0/9 unit 0 family bridge
 vlan-id-list 1-5
user@ce2# set logical-systems CE-2 interfaces ge-1/1/6 unit 0 description
"CONNECTED TO ToR11"
user@ce2# set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge
 interface-mode trunk
user@ce2# set logical-systems CE-2 interfaces ge-1/1/6 unit 0 family bridge
 vlan-id-list 1-5
user@ce2# set logical-systems CE-2 bridge-domains BD-1 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-1 vlan-id 1
user@ce2# set logical-systems CE-2 bridge-domains BD-2 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-2 vlan-id 2
user@ce2# set logical-systems CE-2 bridge-domains BD-3 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-3 vlan-id 3
user@ce2# set logical-systems CE-2 bridge-domains BD-4 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-4 vlan-id 4
user@ce2# set logical-systems CE-2 bridge-domains BD-5 domain-type bridge
user@ce2# set logical-systems CE-2 bridge-domains BD-5 vlan-id 5
```

3. Configure trace options for the interfaces to enable trace logs.

```
[edit]
```

```
user@ce2# set interfaces traceoptions file RO-DCD.log
user@ce2# set interfaces traceoptions file size 10m
user@ce2# set interfaces traceoptions flag all
```

4. Set the number of aggregated Ethernet interfaces.

```
[edit]
```

```
user@ToR11# set chassis aggregated-devices ethernet device-count 1
```

5. Configure the interfaces on the ToR11 device to connect to the MX12, CE-2, CE-1, ToR12, and MX11 devices to enable underlay connectivity.

```
[edit]
```

```
user@ToR11# set interfaces ge-1/0/5 unit 0 description "CONNECTED TO MX12"
user@ToR11# set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.1/24
user@ToR11# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-2"
user@ToR11# set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
user@ToR11# set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
user@ToR11# set interfaces ge-1/1/0 description "CONNECTED TO CE-1"
user@ToR11# set interfaces ge-1/1/0 gigether-options 802.3ad ae0
user@ToR11# set interfaces ge-1/1/1 unit 0 description "CONNECTED TO MX11"
user@ToR11# set interfaces ge-1/1/1 unit 0 family inet address 192.168.3.1/24
user@ToR11# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR12"
user@ToR11# set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.1/24
```

6. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-1 end host device. The ESI value is globally unique across the entire EVPN domain. The all-active configuration enables ToR11 and ToR12 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```
[edit]
```

```
user@ToR11# set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
user@ToR11# set interfaces ae0 esi all-active
user@ToR11# set interfaces ae0 aggregated-ether-options lacp active
user@ToR11# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@ToR11# set interfaces ae0 aggregated-ether-options lacp system-id
11:11:11:11:11:11
```

```
user@ToR11# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR11# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
```

7. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@ToR11# set interfaces lo0 unit 81 family inet address 192.0.2.11/32
user@ToR11# set routing-options router-id 192.0.2.11
user@ToR11# set routing-options autonomous-system 100
```

8. Configure load balancing on ToR11.

```
[edit]
```

```
user@ToR11# set routing-options forwarding-table export evpn-pplb
```

9. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR11 and ToR12) and gateway routers (MX11 and MX12).

```
[edit]
```

```
user@ToR11# set protocols bgp local-as 100
user@ToR11# set protocols bgp group MX11 type external
user@ToR11# set protocols bgp group MX11 local-address 192.168.3.1
user@ToR11# set protocols bgp group MX11 export LO
user@ToR11# set protocols bgp group MX11 export TEST
user@ToR11# set protocols bgp group MX11 peer-as 400
user@ToR11# set protocols bgp group MX11 neighbor 192.168.3.2 family inet
unicast
user@ToR11# set protocols bgp group MX12 type external
user@ToR11# set protocols bgp group MX12 local-address 192.168.4.1
user@ToR11# set protocols bgp group MX12 export LO
user@ToR11# set protocols bgp group MX12 export TEST
user@ToR11# set protocols bgp group MX12 peer-as 500
user@ToR11# set protocols bgp group MX12 neighbor 192.168.4.2 family inet
unicast
user@ToR11# set protocols bgp group ToR12 type external
user@ToR11# set protocols bgp group ToR12 local-address 192.168.2.1
user@ToR11# set protocols bgp group ToR12 export LO
user@ToR11# set protocols bgp group ToR12 export TEST
user@ToR11# set protocols bgp group ToR12 peer-as 200
user@ToR11# set protocols bgp group ToR12 local-as 100
```

```
user@ToR11# set protocols bgp group ToR12 neighbor 192.168.2.2 family inet
unicast
```

10. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR11 and ToR12) and gateway routers (MX11 and MX12) and set EVPN as the signaling protocol.

    a. Configure a MP-EBGP overlay to connect between ToR11 and MX11 using EVPN signaling.

    ```
    [edit]
    ```

    ```
    user@ToR11# set protocols bgp group MX11-EVPN type external
    user@ToR11# set protocols bgp group MX11-EVPN multihop ttl 2
    user@ToR11# set protocols bgp group MX11-EVPN multihop no-nexthop-change
    user@ToR11# set protocols bgp group MX11-EVPN local-address 192.0.2.11
    user@ToR11# set protocols bgp group MX11-EVPN export TEST
    user@ToR11# set protocols bgp group MX11-EVPN peer-as 400
    user@ToR11# set protocols bgp group MX11-EVPN local-as 100
    user@ToR11# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family
     evpn signaling
    ```

    b. Configure a MP-EBGP overlay to connect between ToR11 and MX12 using EVPN signaling.

    ```
    [edit]
    ```

    ```
    user@ToR11# set protocols bgp group MX12-EVPN type external
    user@ToR11# set protocols bgp group MX12-EVPN multihop ttl 2
    user@ToR11# set protocols bgp group MX12-EVPN multihop no-nexthop-change
    user@ToR11# set protocols bgp group MX12-EVPN local-address 192.0.2.11
    user@ToR11# set protocols bgp group MX12-EVPN export TEST
    user@ToR11# set protocols bgp group MX12-EVPN peer-as 500
    user@ToR11# set protocols bgp group MX12-EVPN local-as 100
    user@ToR11# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family
     evpn signaling
    ```

    c. Configure a MP-EBGP overlay to connect between ToR11 and ToR12 using EVPN signaling.

    ```
    [edit]
    ```

    ```
    user@ToR11# set protocols bgp group ToR12-EVPN type external
    ```

```
user@ToR11# set protocols bgp group ToR12-EVPN multihop ttl 2
user@ToR11# set protocols bgp group ToR12-EVPN multihop no-nexthop-change
user@ToR11# set protocols bgp group ToR12-EVPN local-address 192.0.2.11
user@ToR11# set protocols bgp group ToR12-EVPN export TEST
user@ToR11# set protocols bgp group ToR12-EVPN peer-as 200
user@ToR11# set protocols bgp group ToR12-EVPN local-as 100
user@ToR11# set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family
 evpn signaling
```

11. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@ToR11# set protocols l2-learning traceoptions file TOR11-L2ALD.log
user@ToR11# set protocols l2-learning traceoptions file size 10m
user@ToR11# set protocols l2-learning traceoptions level all
user@ToR11# set protocols l2-learning traceoptions flag all
```

12. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@ToR11# set policy-options policy-statement LO term 1 from protocol
direct
user@ToR11# set policy-options policy-statement LO term 1 from route-filter
 192.0.2.11/32 exact
user@ToR11# set policy-options policy-statement LO term 1 then accept
```

13. Configure community policy options.

```
[edit]
```

```
user@ToR11# set policy-options community NO-EXPORT members no-advertise
user@ToR11# set policy-options community NO-EXPORT members no-export
user@ToR11# set policy-options community NO-EXPORT members no-export-subconfed
```

14. Apply load balance.

```
[edit]
```

```
user@ToR11# set policy-options policy-statement TEST then community add
NO-EXPORT
user@ToR11# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR11# set policy-options policy-statement evpn-pplb then load-balance
 per-packet
```

15. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@ToR11# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.81
user@ToR11# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR11# set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
user@ToR11# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR11# set routing-instances EVPN-VXLAN-1 route-distinguisher
192.0.2.11:1
user@ToR11# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file TOR11-EVPN-VXLAN-1.log
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
 vxlan
user@ToR11# set routing-instances EVPN-VXLAN-1 protocols evpn
extended-vni-list 1-5
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id
1
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id
2
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id
3
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
 bridge
```

```
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id
4
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id
5
user@ToR11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
```

Configuring ToR12

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router ToR12:

1. Set the system hostname.

```
[edit]
```

```
user@ToR12# set system host-name ToR12
```

2. Configure the interfaces and bridge domains on the CE-1 device to enable Layer 2 connectivity.

```
[edit]
```

```
user@ce1# set logical-systems CE-1 interfaces ge-1/0/9 unit 0 description
"CONNECTED TO Host 1"
user@ce1# set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge
 interface-mode trunk
user@ce1# set logical-systems CE-1 interfaces ge-1/0/9 unit 0 family bridge
 vlan-id-list 1-5
user@ce1# set logical-systems CE-1 interfaces ae1 unit 0 description
"CONNECTED TO ToR12"
user@ce1# set logical-systems CE-1 interfaces ae1 unit 0 family bridge
interface-mode trunk
user@ce1# set logical-systems CE-1 interfaces ae1 unit 0 family bridge
vlan-id-list 1-5
user@ce1# set logical-systems CE-1 bridge-domains BD-1 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-1 vlan-id 1
user@ce1# set logical-systems CE-1 bridge-domains BD-2 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-2 vlan-id 2
user@ce1# set logical-systems CE-1 bridge-domains BD-3 domain-type bridge
```

```
user@ce1# set logical-systems CE-1 bridge-domains BD-3 vlan-id 3
user@ce1# set logical-systems CE-1 bridge-domains BD-4 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-4 vlan-id 4
user@ce1# set logical-systems CE-1 bridge-domains BD-5 domain-type bridge
user@ce1# set logical-systems CE-1 bridge-domains BD-5 vlan-id 5
```

3. Configure the interfaces and bridge domains on the CE-3 device to enable Layer 2 connectivity.

```
[edit]
```

```
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 description
"CONNECTED TO ToR12"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge
 interface-mode trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge
 vlan-id-list 1-5
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 description
"CONNECTED TO Host 3"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge
 interface-mode trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge
 vlan-id-list 1-5
user@ce3# set logical-systems CE-3 bridge-domains BD-1 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-1 vlan-id 1
user@ce3# set logical-systems CE-3 bridge-domains BD-2 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-2 vlan-id 2
user@ce3# set logical-systems CE-3 bridge-domains BD-3 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-3 vlan-id 3
user@ce3# set logical-systems CE-3 bridge-domains BD-4 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-4 vlan-id 4
user@ce3# set logical-systems CE-3 bridge-domains BD-5 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-5 vlan-id 5
```

4. Configure trace options for the interfaces to enable trace logs.

```
[edit]
```

```
user@ce3# set interfaces traceoptions file R1-DCD.log
user@ce3# set interfaces traceoptions file size 10m
user@ce3# set interfaces traceoptions flag all
```

5. Set the number of aggregated Ethernet interfaces.

```
[edit]
```

```
user@ToR12# set chassis aggregated-devices ethernet device-count 2
```

6. Configure the interfaces on the ToR12 device to connect to the MX12, CE-2, CE-3, ToR11, and MX11 devices to enable underlay connectivity.

```
[edit]
```

```
user@ToR12# set interfaces ge-1/0/0 unit 0 description "CONNECTED TO  MX11"
user@ToR12# set interfaces ge-1/0/0 unit 0 family inet address 192.168.6.1/24
user@ToR12# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO MX12"
user@ToR12# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.1/24
user@ToR12# set interfaces ge-1/0/6 description "CONNECTED TO CE-1"
user@ToR12# set interfaces ge-1/0/6 gigether-options 802.3ad ae0
user@ToR12# set interfaces ge-1/0/7 unit 0 description "CONNECTED TO CE-3"
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ge-1/1/0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/0 gigether-options 802.3ad ae1
user@ToR12# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.2/24
user@ToR12# set interfaces ge-1/1/6 description "CONNECTED TO ToR12"
user@ToR12# set interfaces ge-1/1/6 gigether-options 802.3ad ae1
```

7. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-1 end host device. The ESI value is globally unique across the entire EVPN domain. The all-active configuration enables ToR11 and ToR12 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```
[edit]
```

```
user@ToR12# set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
user@ToR12# set interfaces ae0 esi all-active
user@ToR12# set interfaces ae0 aggregated-ether-options lacp system-id
11:11:11:11:11:11
user@ToR12# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ae1 aggregated-ether-options lacp active
user@ToR12# set interfaces ae1 aggregated-ether-options lacp periodic fast
```

8. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@ToR12# set interfaces lo0 unit 82 family inet address 192.0.2.12/32
user@ToR12# set routing-options router-id 192.0.2.12
user@ToR12# set routing-options autonomous-system 200
```

9. Configure load balancing on ToR12.

```
[edit]
```

```
user@ToR12# set routing-options forwarding-table export evpn-pplb
```

10. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12).

```
[edit]
```

```
user@ToR12# set protocols bgp local-as 200
user@ToR12# set protocols bgp group MX11 type external
user@ToR12# set protocols bgp group MX11 local-address 192.168.6.1
user@ToR12# set protocols bgp group MX11 export LO
user@ToR12# set protocols bgp group MX11 export TEST
user@ToR12# set protocols bgp group MX11 peer-as 400
user@ToR12# set protocols bgp group MX11 local-as 200
user@ToR12# set protocols bgp group MX11 neighbor 192.168.6.2 family inet
unicast
user@ToR12# set protocols bgp group MX12 type external
user@ToR12# set protocols bgp group MX12 local-address 192.168.5.1
user@ToR12# set protocols bgp group MX12 export LO
user@ToR12# set protocols bgp group MX12 export TEST
user@ToR12# set protocols bgp group MX12 peer-as 500
user@ToR12# set protocols bgp group MX12 local-as 200
user@ToR12# set protocols bgp group MX12 neighbor 192.168.5.2 family inet
unicast
user@ToR12# set protocols bgp group ToR11 type external
user@ToR12# set protocols bgp group ToR11 local-address 192.168.2.2
user@ToR12# set protocols bgp group ToR11 export LO
user@ToR12# set protocols bgp group ToR11 export TEST
user@ToR12# set protocols bgp group ToR11 peer-as 100
user@ToR12# set protocols bgp group ToR11 local-as 200
user@ToR12# set protocols bgp group ToR11 neighbor 192.168.2.1 family inet
unicast
```

11. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12) and set EVPN as the signaling protocol.

    a. Configure a MP-EBGP overlay to connect between ToR12 and MX11 using EVPN signaling.

```
[edit]
```

```
user@ToR12# set protocols bgp group MX11-EVPN type external
user@ToR12# set protocols bgp group MX11-EVPN multihop ttl 2
user@ToR12# set protocols bgp group MX11-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group MX11-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group MX11-EVPN export TEST
user@ToR12# set protocols bgp group MX11-EVPN peer-as 400
user@ToR12# set protocols bgp group MX11-EVPN local-as 200
user@ToR12# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family
 evpn signaling
```

b. Configure a MP-EBGP overlay to connect between ToR12 and MX12 using EVPN signaling.

```
[edit]
```

```
user@ToR12# set protocols bgp group MX12-EVPN type external
user@ToR12# set protocols bgp group MX12-EVPN multihop ttl 2
user@ToR12# set protocols bgp group MX12-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group MX12-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group MX12-EVPN export TEST
user@ToR12# set protocols bgp group MX12-EVPN peer-as 500
user@ToR12# set protocols bgp group MX12-EVPN local-as 200
user@ToR12# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family
 evpn signaling
```

c. Configure a MP-EBGP overlay to connect between ToR12 and ToR11 using EVPN signaling.

```
[edit]
```

```
user@ToR12# set protocols bgp group ToR11-EVPN type external
user@ToR12# set protocols bgp group ToR11-EVPN multihop ttl 2
user@ToR12# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group ToR11-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group ToR11-EVPN export TEST
user@ToR12# set protocols bgp group ToR11-EVPN peer-as 100
user@ToR12# set protocols bgp group ToR11-EVPN local-as 200
user@ToR12# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family
 evpn signaling
user@ToR12# set protocols bgp group ToR12-EVPN export TEST
```

12. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@ToR12# set protocols l2-learning traceoptions file TOR12-L2ALD.log
user@ToR12# set protocols l2-learning traceoptions file size 10m
user@ToR12# set protocols l2-learning traceoptions level all
user@ToR12# set protocols l2-learning traceoptions flag all
```

13. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@ToR12# set policy-options policy-statement LO term 1 from protocol
direct
user@ToR12# set policy-options policy-statement LO term 1 from route-filter
 192.0.2.12/32 exact
user@ToR12# set policy-options policy-statement LO term 1 then accept
```

14. Configure community policy options.

```
[edit]
```

```
user@ToR12# set policy-options community NO-EXPORT members no-advertise
user@ToR12# set policy-options community NO-EXPORT members no-export
user@ToR12# set policy-options community NO-EXPORT members no-export-subconfed
```

15. Apply load balance.

```
[edit]
```

```
user@ToR12# set policy-options policy-statement TEST then community add
NO-EXPORT
user@ToR12# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR12# set policy-options policy-statement evpn-pplb then load-balance
 per-packet
```

16. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM

traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@ToR12# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.82
user@ToR12# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ge-1/0/7.0
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR12# set routing-instances EVPN-VXLAN-1 route-distinguisher
192.0.2.12:1
user@ToR12# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file TOR12-EVPN-VXLAN-1.log
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
 vxlan
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn
extended-vni-list 1-5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id
1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id
2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id
3
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id
4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id
5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
```

Configuring Data Center Gateway and WAN Edge 1 Router (MX11)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX11:

1. Set the system hostname.

```
[edit]
```

```
user@MX11# set system host-name MX11
```

2. Configure the interfaces on the MX11 router (DC GW/WAN Edge1) to enable the underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
[edit]
```

```
user@MX11# set interfaces ge-5/1/0 unit 0 description "CONNECTED TO MX12"
user@MX11# set interfaces ge-5/1/0 unit 0 family inet address 192.168.7.1/24
user@MX11# set interfaces ge-5/1/1 unit 0 description "CONNECTED TO ToR11"
user@MX11# set interfaces ge-5/1/1 unit 0 family inet address 192.168.3.2/24
user@MX11# set interfaces ge-5/1/8 unit 0 description "CONNECTED TO ToR12"
user@MX11# set interfaces ge-5/1/8 unit 0 family inet address 192.168.6.2/24
user@MX11# set interfaces ge-5/1/9 unit 0 description "CONNECTED TO P"
user@MX11# set interfaces ge-5/1/9 unit 0 family inet address 203.0.1.1/24
user@MX11# set interfaces ge-5/1/9 unit 0 family mpls
```

3. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12).

```
[edit]
```

```
user@MX11# set protocols bgp group ToR11 type external
user@MX11# set protocols bgp group ToR11 local-address 192.168.3.2
user@MX11# set protocols bgp group ToR11 import TEST
user@MX11# set protocols bgp group ToR11 export TEST
user@MX11# set protocols bgp group ToR11 export LO
user@MX11# set protocols bgp group ToR11 peer-as 100
user@MX11# set protocols bgp group ToR11 local-as 400
user@MX11# set protocols bgp group ToR11 neighbor 192.168.3.1 family inet
```

```
unicast
user@MX11# set protocols bgp group ToR12 type external
user@MX11# set protocols bgp group ToR12 local-address 192.168.6.2
user@MX11# set protocols bgp group ToR12 export TEST
user@MX11# set protocols bgp group ToR12 export LO
user@MX11# set protocols bgp group ToR12 peer-as 200
user@MX11# set protocols bgp group ToR12 local-as 400
user@MX11# set protocols bgp group ToR12 neighbor 192.168.6.1 family inet
unicast
user@MX11# set protocols bgp group MX12 type external
user@MX11# set protocols bgp group MX12 local-address 192.168.7.1
user@MX11# set protocols bgp group MX12 export TEST
user@MX11# set protocols bgp group MX12 export LO
user@MX11# set protocols bgp group MX12 peer-as 500
user@MX11# set protocols bgp group MX12 local-as 400
user@MX11# set protocols bgp group MX12 neighbor 192.168.7.2 family inet
unicast
```

4. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12) and set EVPN as the signaling protocol.

```
[edit]
```

```
user@MX11# set protocols bgp group ToR11-EVPN type external
user@MX11# set protocols bgp group ToR11-EVPN multihop ttl 2
user@MX11# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@MX11# set protocols bgp group ToR11-EVPN local-address 192.0.2.21
user@MX11# set protocols bgp group ToR11-EVPN export TEST
user@MX11# set protocols bgp group ToR11-EVPN peer-as 100
user@MX11# set protocols bgp group ToR11-EVPN local-as 400
user@MX11# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family
evpn signaling
user@MX11# set protocols bgp group ToR12-EVPN type external
user@MX11# set protocols bgp group ToR12-EVPN multihop ttl 2
user@MX11# set protocols bgp group ToR12-EVPN multihop no-nexthop-change
user@MX11# set protocols bgp group ToR12-EVPN local-address 192.0.2.21
user@MX11# set protocols bgp group ToR12-EVPN export TEST
user@MX11# set protocols bgp group ToR12-EVPN peer-as 200
user@MX11# set protocols bgp group ToR12-EVPN local-as 400
user@MX11# set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family
evpn signaling
user@MX11# set protocols bgp group MX12-EVPN type external
user@MX11# set protocols bgp group MX12-EVPN multihop ttl 2
user@MX11# set protocols bgp group MX12-EVPN multihop no-nexthop-change
user@MX11# set protocols bgp group MX12-EVPN local-address 192.0.2.21
user@MX11# set protocols bgp group MX12-EVPN export TEST
user@MX11# set protocols bgp group MX12-EVPN peer-as 500
user@MX11# set protocols bgp group MX12-EVPN local-as 400
user@MX11# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family evpn
 signaling
```

5. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

   a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

   ```
   [edit]
   ```

   ```
   user@MX11# set interfaces irb unit 1 proxy-macip-advertisement
   user@MX11# set interfaces irb unit 1 virtual-gateway-esi
   00:11:aa:aa:aa:aa:aa:aa:aa:aa
   user@MX11# set interfaces irb unit 1 virtual-gateway-esi all-active
   user@MX11# set interfaces irb unit 1 family inet address 10.11.1.12/24
   virtual-gateway-address 10.11.1.10
   ```

   b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

   ```
   [edit]
   ```

   ```
   user@MX11# set interfaces irb unit 2 proxy-macip-advertisement
   user@MX11# set interfaces irb unit 2 virtual-gateway-esi
   00:11:bb:bb:bb:bb:bb:bb:bb:bb
   user@MX11# set interfaces irb unit 2 virtual-gateway-esi all-active
   user@MX11# set interfaces irb unit 2 family inet address 10.12.1.12/24
   virtual-gateway-address 10.12.1.10
   ```

   c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

   ```
   [edit]
   ```

   ```
   user@MX11# set interfaces irb unit 3 proxy-macip-advertisement
   user@MX11# set interfaces irb unit 3 virtual-gateway-esi
   00:11:cc:cc:cc:cc:cc:cc:cc:cc
   user@MX11# set interfaces irb unit 3 virtual-gateway-esi all-active
   user@MX11# set interfaces irb unit 3 family inet address 10.13.1.12/24
   virtual-gateway-address 10.13.1.10
   ```

   d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]
```

```
user@MX11# set interfaces irb unit 4 proxy-macip-advertisement
user@MX11# set interfaces irb unit 4 virtual-gateway-esi
00:11:dd:dd:dd:dd:dd:dd:dd:dd
user@MX11# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX11# set interfaces irb unit 4 family inet address 10.14.1.12/24
virtual-gateway-address 10.14.1.10
```

e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```
[edit]
```

```
user@MX11# set interfaces irb unit 5 proxy-macip-advertisement
user@MX11# set interfaces irb unit 5 virtual-gateway-esi
00:11:ee:ee:ee:ee:ee:ee:ee:ee
user@MX11# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX11# set interfaces irb unit 5 family inet address 10.15.1.12/24
virtual-gateway-address 10.15.1.10
```

6. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@MX11# set protocols l2-learning traceoptions file MX11-L2ALD.log
user@MX11# set protocols l2-learning traceoptions file size 10m
user@MX11# set protocols l2-learning traceoptions level all
user@MX11# set protocols l2-learning traceoptions flag all
```

7. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@MX11# set policy-options policy-statement LO term 1 from protocol direct
user@MX11# set policy-options policy-statement LO term 1 from route-filter
192.0.2.21/32 exact
user@MX11# set policy-options policy-statement LO term 1 then accept
user@MX11# set policy-options policy-statement LO from protocol direct
```

```
user@MX11# set policy-options policy-statement LO from route-filter
192.0.2.21/32 exact
user@MX11# set policy-options policy-statement LO then accept
```

8. Configure community policy options.

```
[edit]
```

```
user@MX11# set policy-options community NO-EXPORT members no-advertise
user@MX11# set policy-options community NO-EXPORT members no-export
user@MX11# set policy-options community NO-EXPORT members no-export-subconfed
```

9. Apply load balance.

```
[edit]
```

```
user@MX11# set policy-options policy-statement TEST then community add
NO-EXPORT
user@MX11# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX11# set policy-options policy-statement evpn-pplb then load-balance
per-packet
```

10. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
[edit]
```

```
user@MX11# set interfaces lt-5/1/0 esi 00:22:22:22:22:22:22:22:22:22
```

11. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```
[edit]
```

```
user@MX11# set interfaces lt-5/1/0 esi all-active
```

12. Configure a pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the

MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]
```

```
user@MX11# set interfaces lt-5/1/0 unit 0 peer-unit 1
user@MX11# set interfaces lt-5/1/0 unit 0 family bridge interface-mode trunk
user@MX11# set interfaces lt-5/1/0 unit 0 family bridge vlan-id-list 1-5
user@MX11# set interfaces lt-5/1/0 unit 1 peer-unit 0
user@MX11# set interfaces lt-5/1/0 unit 1 family bridge interface-mode trunk
user@MX11# set interfaces lt-5/1/0 unit 1 family bridge vlan-id-list 1-5
```

13. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@MX11# set interfaces lo0 unit 84 family inet address 192.0.2.21/32
user@MX11# set interfaces lo0 unit 84 family mpls
user@MX11# set routing-options router-id 192.0.2.21
user@MX11# set routing-options autonomous-system 300
```

14. Configure load balancing on MX11.

```
[edit]
```

```
user@MX11# set routing-options forwarding-table export evpn-pplb
```

15. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX12, P, MX21, MX22).

```
[edit]
```

```
user@MX11# set protocols rsvp interface all
user@MX11# set protocols rsvp interface fxp0.0 disable
user@MX11# set protocols mpls label-switched-path MX11-TO-MX12 to 192.0.2.22
user@MX11# set protocols mpls label-switched-path MX11-TO-P to 203.0.113.1
user@MX11# set protocols mpls label-switched-path MX11-TO-MX21 to
198.51.100.21
```

```
user@MX11# set protocols mpls label-switched-path MX11-TO-MX22 to
198.51.100.22
user@MX11# set protocols mpls interface all
user@MX11# set protocols mpls interface fxp0.0 disable
user@MX11# set protocols bgp local-address 192.0.2.21
user@MX11# set protocols bgp local-as 300
user@MX11# set protocols bgp group INT type internal
user@MX11# set protocols bgp group INT local-address 192.0.2.21
user@MX11# set protocols bgp group INT family evpn signaling
user@MX11# set protocols bgp group INT export TEST
user@MX11# set protocols bgp group INT neighbor 203.0.113.1
user@MX11# set protocols ospf traffic-engineering
user@MX11# set protocols ospf area 0.0.0.0 interface ge-5/1/9.0
user@MX11# set protocols ospf area 0.0.0.0 interface lo0.84 passive
```

16. Configure EVPN-based MPLS routing instances on the MX11 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

```
[edit]
```

```
user@MX11# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX11# set routing-instances EVPN-MPLS-1 interface lt-5/1/0.0
user@MX11# set routing-instances EVPN-MPLS-1 route-distinguisher
192.0.2.21:100
user@MX11# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file MX11-EVPN-MPLS-1.log
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file size 10m
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
flag all
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list
 1-5
user@MX11# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
 no-gateway-community
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type
 bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type
 bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type
 bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type
 bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type
```

```
 bridge
user@MX11# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
```

17. Configure EVPN-VXLAN routing instances on the MX11 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@MX11# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.84
user@MX11# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX11# set routing-instances EVPN-VXLAN-1 interface lt-5/1/0.1
user@MX11# set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.21:1
user@MX11# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file MX11-EVPN-VXLAN-1.log
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
vxlan
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list
 1-5
user@MX11# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
 no-gateway-community
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1
routing-interface irb.1
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2
routing-interface irb.2
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3
routing-interface irb.3
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
 bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
```

```
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4
routing-interface irb.4
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5
routing-interface irb.5
user@MX11# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
user@MX11# set routing-instances VRF instance-type vrf
user@MX11# set routing-instances VRF interface irb.1
user@MX11# set routing-instances VRF interface irb.2
user@MX11# set routing-instances VRF interface irb.3
user@MX11# set routing-instances VRF interface irb.4
user@MX11# set routing-instances VRF interface irb.5
user@MX11# set routing-instances VRF route-distinguisher 1:1
user@MX11# set routing-instances VRF vrf-target target:10:10
```

Configuring Data Center Gateway and WAN Edge 2 Router (MX12)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX12:

1. Set the system hostname.

```
[edit]
```

```
user@MX12# set system host-name MX12
```

2. Configure the P device as the logical system of MX12 data center gateway and WAN edge router.

   a. Configure the P device to operate in the enhanced-ip mode because the EVPN active-active functionality is supported on routers with MPCs and MIC interfaces only. A system reboot is required on committing this configuration.

```
[edit chassis]
```

```
user@P# set network-services enhanced-ip
```

   b. Configure the interfaces of the P device.

```
[edit]
```

```
user@P# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO ToR12"
user@P# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.2/24
user@P# set interfaces ge-1/0/5 unit 0 description "CONNECTED TO TOR11"
user@P# set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.2/24
user@P# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO P"
user@P# set interfaces ge-1/0/6 unit 0 family inet address 203.0.2.1/24
user@P# set interfaces ge-1/0/6 unit 0 family mpls
user@P# set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX11"
user@P# set interfaces ge-1/1/0 unit 0 family inet address 192.168.7.2/24
```

c. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces of the P device. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX12, MX21, MX22).

```
[edit]
```

```
user@P# set logical-systems P protocols rsvp interface all
user@P# set logical-systems P protocols mpls label-switched-path P-TO-MX11
 from 203.0.113.1
user@P# set logical-systems P protocols mpls label-switched-path P-TO-MX11
 to 192.0.2.21
user@P# set logical-systems P protocols mpls label-switched-path P-TO-MX12
 to 192.0.2.22
user@P# set logical-systems P protocols mpls label-switched-path P-TO-MX21
 to 198.51.100.21
user@P# set logical-systems P protocols mpls label-switched-path P-TO-MX22
 to 198.51.100.22
user@P# set logical-systems P protocols mpls interface all
user@P# set logical-systems P protocols bgp local-address 203.0.113.1
user@P# set logical-systems P protocols bgp local-as 300
user@P# set logical-systems P protocols bgp group INT type internal
user@P# set logical-systems P protocols bgp group INT import
BLOCK-VXLAN-ROUTES-FROM-CORE
user@P# set logical-systems P protocols bgp group INT family evpn signaling
user@P# set logical-systems P protocols bgp group INT cluster 203.0.113.1
user@P# set logical-systems P protocols bgp group INT neighbor 192.0.2.21
user@P# set logical-systems P protocols bgp group INT neighbor 192.0.2.22
user@P# set logical-systems P protocols bgp group INT neighbor
198.51.100.21
user@P# set logical-systems P protocols bgp group INT neighbor
198.51.100.22
user@P# set logical-systems P protocols ospf traffic-engineering
user@P# set logical-systems P protocols ospf area 0.0.0.0 interface all
user@P# set logical-systems P protocols ospf area 0.0.0.0 interface lo0.86
```

d. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@P# set logical-systems P interfaces lo0 unit 86 family inet address
 203.0.113.1/32
user@P# set logical-systems P interfaces lo0 unit 86 family mpls
user@P# set logical-systems P routing-options router-id 203.0.113.1
user@P# set logical-systems P routing-options autonomous-system 300
```

e. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 1 from protocol bgp
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 1 from community RT-CORE
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 1 then accept
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 2 from protocol bgp
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 2 from community RT-DC1
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 2 then reject
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 3 from protocol bgp
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 3 from community RT-DC2
user@P# set logical-systems P policy-options policy-statement
BLOCK-VXLAN-ROUTES-FROM-CORE term 3 then reject
```

f. Configure community policy options.

```
[edit]
```

```
user@P# set logical-systems P policy-options community RT-CORE members
target:1:2
user@P# set logical-systems P policy-options community RT-DC1 members
target:1:1
user@P# set logical-systems P policy-options community RT-DC2 members
target:1:3
```

g. Configure trace options for the interfaces to enable trace logs.

```
[edit]
```

```
user@P# set interfaces traceoptions file R3-DCD.log
user@P# set interfaces traceoptions file size 10m
user@P# set interfaces traceoptions flag all
```

3. Configure the interfaces on the MX12 router (DC GW/WAN Edge 2) to enable the underlay connectivity to the MX11, ToR12, ToR11, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
[edit]
```

```
user@MX12# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO ToR12"
user@MX12# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.2/24
user@MX12# set interfaces ge-1/0/5 unit 0 description "CONNECTED TO TOR11"
user@MX12# set interfaces ge-1/0/5 unit 0 family inet address 192.168.4.2/24
user@MX12# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO P"
user@MX12# set interfaces ge-1/0/6 unit 0 family inet address 203.0.2.1/24
user@MX12# set interfaces ge-1/0/6 unit 0 family mpls
user@MX12# set interfaces ge-1/1/0 unit 0 description "CONNECTED TO MX11"
user@MX12# set interfaces ge-1/1/0 unit 0 family inet address 192.168.7.2/24
```

4. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12).

```
[edit]
```

```
user@MX12# set protocols bgp group ToR11 type external
user@MX12# set protocols bgp group ToR11 local-address 192.168.4.2
user@MX12# set protocols bgp group ToR11 export TEST
user@MX12# set protocols bgp group ToR11 export LO
user@MX12# set protocols bgp group ToR11 peer-as 100
user@MX12# set protocols bgp group ToR11 local-as 500
user@MX12# set protocols bgp group ToR11 neighbor 192.168.4.1 family inet
unicast
user@MX12# set protocols bgp group ToR12 type external
user@MX12# set protocols bgp group ToR12 local-address 192.168.5.2
user@MX12# set protocols bgp group ToR12 export TEST
user@MX12# set protocols bgp group ToR12 export LO
user@MX12# set protocols bgp group ToR12 peer-as 200
user@MX12# set protocols bgp group ToR12 local-as 500
user@MX12# set protocols bgp group ToR12 neighbor 192.168.5.1 family inet
unicast
user@MX12# set protocols bgp group MX11 type external
user@MX12# set protocols bgp group MX11 local-address 192.168.7.2
user@MX12# set protocols bgp group MX11 export TEST
```

```
user@MX12# set protocols bgp group MX11 export LO
user@MX12# set protocols bgp group MX11 peer-as 400
user@MX12# set protocols bgp group MX11 local-as 500
user@MX12# set protocols bgp group MX11 neighbor 192.168.7.1 family inet
unicast
```

5. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX11 and MX12) and ToR (ToR11 and ToR12) and set EVPN as the signaling protocol.

```
[edit]
```

```
user@MX12# set protocols bgp group ToR11-EVPN type external
user@MX12# set protocols bgp group ToR11-EVPN multihop ttl 2
user@MX12# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@MX12# set protocols bgp group ToR11-EVPN local-address 192.0.2.22
user@MX12# set protocols bgp group ToR11-EVPN export TEST
user@MX12# set protocols bgp group ToR11-EVPN peer-as 100
user@MX12# set protocols bgp group ToR11-EVPN local-as 500
user@MX12# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family
evpn signaling
user@MX12# set protocols bgp group ToR12-EVPN type external
user@MX12# set protocols bgp group ToR12-EVPN multihop ttl 2
user@MX12# set protocols bgp group ToR12-EVPN multihop no-nexthop-change
user@MX12# set protocols bgp group ToR12-EVPN local-address 192.0.2.22
user@MX12# set protocols bgp group ToR12-EVPN export TEST
user@MX12# set protocols bgp group ToR12-EVPN peer-as 200
user@MX12# set protocols bgp group ToR12-EVPN local-as 500
user@MX12# set protocols bgp group ToR12-EVPN neighbor 192.0.2.12 family
evpn signaling
user@MX12# set protocols bgp group MX11-EVPN type external
user@MX12# set protocols bgp group MX11-EVPN multihop ttl 2
user@MX12# set protocols bgp group MX11-EVPN multihop no-nexthop-change
user@MX12# set protocols bgp group MX11-EVPN local-address 192.0.2.22
user@MX12# set protocols bgp group MX11-EVPN export TEST
user@MX12# set protocols bgp group MX11-EVPN peer-as 400
user@MX12# set protocols bgp group MX11-EVPN local-as 500
user@MX12# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family evpn
 signaling
user@MX12# set protocols bgp group MX12-EVPN export TEST
```

6. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

   a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

```
[edit]
```

```
user@MX12# set interfaces irb unit 1 proxy-macip-advertisement
user@MX12# set interfaces irb unit 1 virtual-gateway-esi
00:11:aa:aa:aa:aa:aa:aa:aa:aa
user@MX12# set interfaces irb unit 1 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 1 family inet address 10.11.1.13/24
virtual-gateway-address 10.11.1.10
```

   b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

```
[edit]
```

```
user@MX12# set interfaces irb unit 2 proxy-macip-advertisement
user@MX12# set interfaces irb unit 2 virtual-gateway-esi
00:11:bb:bb:bb:bb:bb:bb:bb:bb
user@MX12# set interfaces irb unit 2 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 2 family inet address 10.12.1.13/24
virtual-gateway-address 10.12.1.10
```

   c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

```
[edit]
```

```
user@MX12# set interfaces irb unit 3 proxy-macip-advertisement
user@MX12# set interfaces irb unit 3 virtual-gateway-esi
00:11:cc:cc:cc:cc:cc:cc:cc:cc
user@MX12# set interfaces irb unit 3 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 3 family inet address 10.13.1.13/24
virtual-gateway-address 10.13.1.10
```

   d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]
```

```
user@MX12# set interfaces irb unit 4 proxy-macip-advertisement
user@MX12# set interfaces irb unit 4 virtual-gateway-esi
00:11:dd:dd:dd:dd:dd:dd:dd:dd
user@MX12# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 4 family inet address 10.14.1.13/24
virtual-gateway-address 10.14.1.10
```

e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```
[edit]
```

```
user@MX12# set interfaces irb unit 5 proxy-macip-advertisement
user@MX12# set interfaces irb unit 5 virtual-gateway-esi
00:11:ee:ee:ee:ee:ee:ee:ee:ee
user@MX12# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX12# set interfaces irb unit 5 family inet address 10.15.1.13/24
virtual-gateway-address 10.15.1.10
```

7. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@MX12# set protocols l2-learning traceoptions file MX12-L2ALD.log
user@MX12# set protocols l2-learning traceoptions file size 10m
user@MX12# set protocols l2-learning traceoptions level all
user@MX12# set protocols l2-learning traceoptions flag all
```

8. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@MX12# set policy-options policy-statement LO from protocol direct
user@MX12# set policy-options policy-statement LO from route-filter
192.0.2.22/32 exact
user@MX12# set policy-options policy-statement LO then accept
user@MX12# set policy-options policy-statement LO term 1 from protocol direct
```

```
user@MX12# set policy-options policy-statement LO term 1 from route-filter
192.0.2.22/32 exact
user@MX12# set policy-options policy-statement LO term 1 then accept
user@MX12# set policy-options policy-statement TEST from protocol bgp
user@MX12# set policy-options policy-statement TEST from protocol evpn
```

9. Configure community policy options.

```
[edit]
```

```
user@MX12# set policy-options community NO-EXPORT members no-advertise
user@MX12# set policy-options community NO-EXPORT members no-export
user@MX12# set policy-options community NO-EXPORT members no-export-subconfed
```

10. Apply load balance.

```
[edit]
```

```
user@MX12# set policy-options policy-statement TEST then community add
NO-EXPORT
user@MX12# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX12# set policy-options policy-statement evpn-pplb then load-balance
per-packet
```

11. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
[edit]
```

```
user@MX12# set interfaces lt-1/0/0 esi 00:22:22:22:22:22:22:22:22:22
```

12. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```
[edit]
```

```
user@MX12# set interfaces lt-1/0/0 esi all-active
```

13. Configure a pair of logical tunnel (lt-) interface on the MX12 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]
```

```
user@MX12# set interfaces lt-1/0/0 unit 0 peer-unit 1
user@MX12# set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
user@MX12# set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5
user@MX12# set interfaces lt-1/0/0 unit 1 peer-unit 0
user@MX12# set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
user@MX12# set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5
```

14. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@MX12# set interfaces lo0 unit 85 family inet address 192.0.2.22/32
user@MX12# set interfaces lo0 unit 85 family mpls
user@MX12# set routing-options router-id 192.0.2.22
user@MX12# set routing-options autonomous-system 300
```

15. Configure load balancing on MX12.

```
[edit]
```

```
user@MX12# set routing-options forwarding-table export evpn-pplb
```

16. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX21, P, MX22).

```
[edit]
```

```
user@MX12# set protocols rsvp interface all
user@MX12# set protocols rsvp interface fxp0.0 disable
user@MX12# set protocols mpls label-switched-path MX12-TO-MX11 to 192.0.2.21
user@MX12# set protocols mpls label-switched-path MX12-TO-P to 203.0.113.1
```

```
user@MX12# set protocols mpls label-switched-path MX12-TO-MX21 to
198.51.100.21
user@MX12# set protocols mpls label-switched-path MX12-TO-MX22 to
198.51.100.22
user@MX12# set protocols mpls interface all
user@MX12# set protocols mpls interface fxp0.0 disable
user@MX12# set protocols bgp local-address 192.0.2.22
user@MX12# set protocols bgp local-as 300
user@MX12# set protocols bgp group INT type internal
user@MX12# set protocols bgp group INT family evpn signaling
user@MX12# set protocols bgp group INT export TEST
user@MX12# set protocols bgp group INT neighbor 203.0.113.1
user@MX12# set protocols ospf traffic-engineering
user@MX12# set protocols ospf area 0.0.0.0 interface ge-1/0/6.0
user@MX12# set protocols ospf area 0.0.0.0 interface lo0.85 passive
```

17. Configure EVPN-based MPLS routing instances on the MX12 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

```
[edit]
```

```
user@MX12# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX12# set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
user@MX12# set routing-instances EVPN-MPLS-1 route-distinguisher
192.0.2.22:100
user@MX12# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file MX12-EVPN-MPLS-1.log
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file size 10m
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
flag all
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list
 1-5
user@MX12# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
 no-gateway-community
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type
 bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type
 bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type
 bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type
 bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type
```

```
 bridge
user@MX12# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
```

18. Configure EVPN-VXLAN routing instances on the MX12 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@MX12# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.85
user@MX12# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX12# set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
user@MX12# set routing-instances EVPN-VXLAN-1 route-distinguisher 192.0.2.22:1
user@MX12# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file MX12-EVPN-VXLAN-1.log
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
vxlan
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list
 1-4
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list
 5
user@MX12# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
 no-gateway-community
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1
routing-interface irb.1
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2
routing-interface irb.2
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3
routing-interface irb.3
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
```

```
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
 bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4
routing-interface irb.4
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5
routing-interface irb.5
user@MX12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
user@MX12# set routing-instances VRF instance-type vrf
user@MX12# set routing-instances VRF interface irb.1
user@MX12# set routing-instances VRF interface irb.2
user@MX12# set routing-instances VRF interface irb.3
user@MX12# set routing-instances VRF interface irb.4
user@MX12# set routing-instances VRF interface irb.5
user@MX12# set routing-instances VRF route-distinguisher 1:1
user@MX12# set routing-instances VRF vrf-target target:10:10
```

Configuring Data Center Gateway and WAN Edge 3 Router (MX21)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX21:

1. Set the system hostname.

```
[edit]
```

```
user@MX21# set system host-name MX21
```

2. Configure the interfaces on the MX21 router (DC GW/WAN Edge 3) to enable the underlay connectivity to the MX22, ToR22, ToR21, and P devices, which is the EVPN-VXLAN part of DC2 network.

```
[edit]
```

```
user@MX21# set interfaces ge-3/0/0 unit 0 description "CONNECTED TO MX22"
user@MX21# set interfaces ge-3/0/0 unit 0 family inet address 192.168.13.1/24
user@MX21# set interfaces ge-3/1/0 unit 0 description "CONNECTED TO ToR22"
```

```
user@MX21# set interfaces ge-3/1/0 unit 0 family inet address 192.168.8.1/24
user@MX21# set interfaces ge-5/0/0 unit 0 description "CONNECTED TO P"
user@MX21# set interfaces ge-5/0/0 unit 0 family inet address 203.0.4.1/24
user@MX21# set interfaces ge-5/0/0 unit 0 family mpls
user@MX21# set interfaces ge-5/0/1 unit 0 description "CONNECTED TO ToR21"
user@MX21# set interfaces ge-5/0/1 unit 0 family inet address 192.168.9.1/24
```

3. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22).

```
[edit]
```

```
user@MX21# set protocols bgp group ToR21 type external
user@MX21# set protocols bgp group ToR21 local-address 192.168.9.1
user@MX21# set protocols bgp group ToR21 export TEST
user@MX21# set protocols bgp group ToR21 export LO
user@MX21# set protocols bgp group ToR21 peer-as 600
user@MX21# set protocols bgp group ToR21 local-as 800
user@MX21# set protocols bgp group ToR21 neighbor 192.168.9.2 family inet
unicast
user@MX21# set protocols bgp group ToR22 type external
user@MX21# set protocols bgp group ToR22 local-address 192.168.8.1
user@MX21# set protocols bgp group ToR22 export TEST
user@MX21# set protocols bgp group ToR22 export LO
user@MX21# set protocols bgp group ToR22 peer-as 700
user@MX21# set protocols bgp group ToR22 local-as 800
user@MX21# set protocols bgp group ToR22 neighbor 192.168.8.2 family inet
unicast
user@MX21# set protocols bgp group MX22 type external
user@MX21# set protocols bgp group MX22 local-address 192.168.13.1
user@MX21# set protocols bgp group MX22 export TEST
user@MX21# set protocols bgp group MX22 export LO
user@MX21# set protocols bgp group MX22 peer-as 900
user@MX21# set protocols bgp group MX22 local-as 800
user@MX21# set protocols bgp group MX22 neighbor 10.115.15.2 family inet
unicast
```

4. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22) and set EVPN as the signaling protocol.

```
[edit]
```

```
user@MX21# set protocols bgp group ToR21-EVPN type external
user@MX21# set protocols bgp group ToR21-EVPN multihop ttl 2
user@MX21# set protocols bgp group ToR21-EVPN multihop no-nexthop-change
user@MX21# set protocols bgp group ToR21-EVPN local-address 198.51.100.21
user@MX21# set protocols bgp group ToR21-EVPN peer-as 600
```

```
user@MX21# set protocols bgp group ToR21-EVPN local-as 800
user@MX21# set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family
 evpn signaling
user@MX21# set protocols bgp group ToR22-EVPN type external
user@MX21# set protocols bgp group ToR22-EVPN multihop ttl 2
user@MX21# set protocols bgp group ToR22-EVPN multihop no-nexthop-change
user@MX21# set protocols bgp group ToR22-EVPN local-address 198.51.100.21
user@MX21# set protocols bgp group ToR22-EVPN peer-as 700
user@MX21# set protocols bgp group ToR22-EVPN local-as 800
user@MX21# set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family
 evpn signaling
user@MX21# set protocols bgp group MX22-EVPN type external
user@MX21# set protocols bgp group MX22-EVPN multihop ttl 2
user@MX21# set protocols bgp group MX22-EVPN multihop no-nexthop-change
user@MX21# set protocols bgp group MX22-EVPN local-address 198.51.100.21
user@MX21# set protocols bgp group MX22-EVPN peer-as 900
user@MX21# set protocols bgp group MX22-EVPN local-as 800
user@MX21# set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family
evpn signaling
```

5. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

   a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

   ```
   [edit]
   ```

   ```
   user@MX21# set interfaces irb unit 1 proxy-macip-advertisement
   user@MX21# set interfaces irb unit 1 virtual-gateway-esi
   00:22:aa:aa:aa:aa:aa:aa:aa:aa
   user@MX21# set interfaces irb unit 1 virtual-gateway-esi all-active
   user@MX21# set interfaces irb unit 1 family inet address 10.11.1.14/24
   virtual-gateway-address 10.11.1.11
   ```

   b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

   ```
   [edit]
   ```

   ```
   user@MX21# set interfaces irb unit 2 proxy-macip-advertisement
   user@MX21# set interfaces irb unit 2 virtual-gateway-esi
   00:22:bb:bb:bb:bb:bb:bb:bb:bb
   user@MX21# set interfaces irb unit 2 virtual-gateway-esi all-active
   user@MX21# set interfaces irb unit 2 family inet address 10.12.1.14/24
   virtual-gateway-address 10.12.1.11
   ```

c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

```
[edit]
```

```
user@MX21# set interfaces irb unit 3 proxy-macip-advertisement
user@MX21# set interfaces irb unit 3 virtual-gateway-esi
00:22:cc:cc:cc:cc:cc:cc:cc:cc
user@MX21# set interfaces irb unit 3 virtual-gateway-esi all-active
user@MX21# set interfaces irb unit 3 family inet address 10.13.1.14/24
virtual-gateway-address 10.13.1.11
```

d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]
```

```
user@MX21# set interfaces irb unit 4 proxy-macip-advertisement
user@MX21# set interfaces irb unit 4 virtual-gateway-esi
00:22:dd:dd:dd:dd:dd:dd:dd:dd
user@MX21# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX21# set interfaces irb unit 4 family inet address 10.14.1.14/24
virtual-gateway-address 10.14.1.11
```

e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```
[edit]
```

```
user@MX21# set interfaces irb unit 5 proxy-macip-advertisement
user@MX21# set interfaces irb unit 5 virtual-gateway-esi
00:22:ee:ee:ee:ee:ee:ee:ee:ee
user@MX21# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX21# set interfaces irb unit 5 family inet address 10.15.1.14/24
virtual-gateway-address 10.15.1.11
```

6. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@MX21# set protocols l2-learning traceoptions file MX21-L2ALD.log
user@MX21# set protocols l2-learning traceoptions file size 10m
user@MX21# set protocols l2-learning traceoptions level all
user@MX21# set protocols l2-learning traceoptions flag all
```

7. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@MX21# set policy-options policy-statement LO from protocol direct
user@MX21# set policy-options policy-statement LO from route-filter
198.51.100.21/32 exact
user@MX21# set policy-options policy-statement LO then accept
user@MX21# set policy-options policy-statement TEST1 term 1 from protocol
bgp
user@MX21# set policy-options policy-statement TEST1 term 1 from external
user@MX21# set policy-options policy-statement TEST1 term 1 then reject
```

8. Configure community policy options.

```
[edit]
```

```
user@MX21# set policy-options community NO-EXPORT members no-advertise
user@MX21# set policy-options community NO-EXPORT members no-export
user@MX21# set policy-options community NO-EXPORT members no-export-subconfed
```

9. Apply load balance.

```
[edit]
```

```
user@MX21# set policy-options policy-statement TEST then community add
NO-EXPORT
user@MX21# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX21# set policy-options policy-statement evpn-pplb then load-balance
per-packet
```

10. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC2 network.

```
[edit]
```

```
user@MX21# set interfaces lt-5/0/0 esi 00:33:33:33:33:33:33:33:33:33
```

11. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```
[edit]
```

```
user@MX21# set interfaces lt-5/0/0 esi all-active
```

12. Configure a pair of logical tunnel (lt-) interface on the MX21 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]
```

```
user@MX21# set interfaces lt-5/0/0 unit 0 peer-unit 1
user@MX21# set interfaces lt-5/0/0 unit 0 family bridge interface-mode trunk
user@MX21# set interfaces lt-5/0/0 unit 0 family bridge vlan-id-list 1-5
user@MX21# set interfaces lt-5/0/0 unit 1 peer-unit 0
user@MX21# set interfaces lt-5/0/0 unit 1 family bridge interface-mode trunk
user@MX21# set interfaces lt-5/0/0 unit 1 family bridge vlan-id-list 1-5
```

13. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@MX21# set interfaces lo0 unit 87 family inet address 198.51.100.21/32
user@MX21# set interfaces lo0 unit 87 family mpls
user@MX21# set routing-options router-id 198.51.100.21
user@MX21# set routing-options autonomous-system 300
```

14. Configure load balancing on MX21.

```
[edit]
```

```
user@MX21# set routing-options forwarding-table export evpn-pplb
```

15. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX12, P, MX22).

```
[edit]
```

```
user@MX21# set protocols rsvp interface all
user@MX21# set protocols rsvp interface fxp0.0 disable
user@MX21# set protocols mpls label-switched-path MX21-TO-MX11 to 192.0.2.21
user@MX21# set protocols mpls label-switched-path MX21-TO-MX12 to 192.0.2.22
user@MX21# set protocols mpls label-switched-path MX21-TO-P to 203.0.113.1
user@MX21# set protocols mpls label-switched-path MX21-TO-MX22 to
198.51.100.22
user@MX21# set protocols mpls interface all
user@MX21# set protocols mpls interface fxp0.0 disable
user@MX21# set protocols bgp local-address 198.51.100.21
user@MX21# set protocols bgp export TEST
user@MX21# set protocols bgp local-as 300
user@MX21# set protocols bgp group INT type internal
user@MX21# set protocols bgp group INT local-address 198.51.100.21
user@MX21# set protocols bgp group INT family evpn signaling
user@MX21# set protocols bgp group INT export TEST1
user@MX21# set protocols bgp group INT neighbor 203.0.113.1
user@MX21# set protocols ospf traffic-engineering
user@MX21# set protocols ospf area 0.0.0.0 interface ge-5/0/0.0
user@MX21# set protocols ospf area 0.0.0.0 interface lo0.87 passive
```

16. Configure EVPN-based MPLS routing instances on the MX21 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

```
[edit]
```

```
user@MX21# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX21# set routing-instances EVPN-MPLS-1 interface lt-5/0/0.0
user@MX21# set routing-instances EVPN-MPLS-1 route-distinguisher
198.51.100.21:100
user@MX21# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file MX21-EVPN-MPLS-1.log
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file size 10m
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
```

```
flag all
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list
 1-5
user@MX21# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
 no-gateway-community
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type
 bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type
 bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type
 bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type
 bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type
 bridge
user@MX21# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
```

17. Configure EVPN-VXLAN routing instances on the MX21 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@MX21# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.87
user@MX21# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX21# set routing-instances EVPN-VXLAN-1 interface lt-5/0/0.1
user@MX21# set routing-instances EVPN-VXLAN-1 route-distinguisher
198.51.100.21:1
user@MX21# set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file MX21-EVPN-VXLAN-1.log
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
vxlan
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list
 1-5
user@MX21# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
 no-gateway-community
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1
```

```
routing-interface irb.1
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2
routing-interface irb.2
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3
routing-interface irb.3
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
 bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4
routing-interface irb.4
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5
routing-interface irb.5
user@MX21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
user@MX21# set routing-instances VRF instance-type vrf
user@MX21# set routing-instances VRF interface irb.1
user@MX21# set routing-instances VRF interface irb.2
user@MX21# set routing-instances VRF interface irb.3
user@MX21# set routing-instances VRF interface irb.4
user@MX21# set routing-instances VRF interface irb.5
user@MX21# set routing-instances VRF route-distinguisher 1:1
user@MX21# set routing-instances VRF vrf-target target:10:10
```

Configuring Data Center Gateway and WAN Edge 4 Router (MX22)

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure an MX Series router as the data center gateway and WAN edge router and name it as MX22:

1. Set the system hostname.

```
[edit]
```

```
user@MX22# set system host-name MX22
```

2. Configure the interfaces on the MX22 router (DC GW/WAN Edge 4) to enable the underlay connectivity to the MX22, ToR21, MX21, and P devices, which is the EVPN-VXLAN part of DC2 network.

```
[edit]
```

```
user@MX22# set interfaces xe-0/0/0 unit 0 description "CONNECTED TO ToR22"
user@MX22# set interfaces xe-0/0/0 unit 0 family inet address 192.168.11.1/24
user@MX22# set interfaces xe-0/0/1 unit 0 description "CONNECTED TO ToR21"
user@MX22# set interfaces xe-0/0/1 unit 0 family inet address 192.168.10.1/24
user@MX22# set interfaces ge-1/0/0 unit 0 description "CONNECTED TO MX21"
user@MX22# set interfaces ge-1/0/0 unit 0 family inet address 10.115.15.2/24
user@MX22# set interfaces ge-1/0/2 unit 0 description "CONNECTED TO P"
user@MX22# set interfaces ge-1/0/2 unit 0 family inet address 203.0.3.1/24
user@MX22# set interfaces ge-1/0/2 unit 0 family mpls
```

3. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22).

```
[edit]
```

```
user@MX22# set protocols bgp group ToR21 type external
user@MX22# set protocols bgp group ToR21 local-address 192.168.10.1
user@MX22# set protocols bgp group ToR21 export TEST
user@MX22# set protocols bgp group ToR21 export LO
user@MX22# set protocols bgp group ToR21 peer-as 600
user@MX22# set protocols bgp group ToR21 local-as 900
user@MX22# set protocols bgp group ToR21 neighbor 10.102.2.1 family inet
unicast
```

```
user@MX22# set protocols bgp group ToR22 type external
user@MX22# set protocols bgp group ToR22 local-address 192.168.11.1
user@MX22# set protocols bgp group ToR22 export TEST
user@MX22# set protocols bgp group ToR22 export LO
user@MX22# set protocols bgp group ToR22 peer-as 700
user@MX22# set protocols bgp group ToR22 local-as 900
user@MX22# set protocols bgp group ToR22 neighbor 192.168.11.2 family inet
unicast
user@MX22# set protocols bgp group MX21 type external
user@MX22# set protocols bgp group MX21 local-address 10.115.15.2
user@MX22# set protocols bgp group MX21 export TEST
user@MX22# set protocols bgp group MX21 export LO
user@MX22# set protocols bgp group MX21 peer-as 800
user@MX22# set protocols bgp group MX21 local-as 900
user@MX22# set protocols bgp group MX21 neighbor 192.168.13.1 family inet
unicast
```

4. Configure a multiprotocol external BGP (MP-EBGP) overlay connectivity between the gateway routers (MX21 and MX22) and ToR (ToR21 and ToR22) and set EVPN as the signaling protocol.

```
[edit]
```

```
user@MX22# set protocols bgp group ToR21-EVPN type external
user@MX22# set protocols bgp group ToR21-EVPN multihop ttl 2
user@MX22# set protocols bgp group ToR21-EVPN multihop no-nexthop-change
user@MX22# set protocols bgp group ToR21-EVPN local-address 198.51.100.22
user@MX22# set protocols bgp group ToR21-EVPN peer-as 600
user@MX22# set protocols bgp group ToR21-EVPN local-as 900
user@MX22# set protocols bgp group ToR21-EVPN neighbor 198.51.100.11 family
 evpn signaling
user@MX22# set protocols bgp group ToR22-EVPN type external
user@MX22# set protocols bgp group ToR22-EVPN multihop ttl 2
user@MX22# set protocols bgp group ToR22-EVPN multihop no-nexthop-change
user@MX22# set protocols bgp group ToR22-EVPN local-address 198.51.100.22
user@MX22# set protocols bgp group ToR22-EVPN peer-as 700
user@MX22# set protocols bgp group ToR22-EVPN local-as 900
user@MX22# set protocols bgp group ToR22-EVPN neighbor 198.51.100.12 family
 evpn signaling
user@MX22# set protocols bgp group MX21-EVPN type external
user@MX22# set protocols bgp group MX21-EVPN multihop ttl 2
user@MX22# set protocols bgp group MX21-EVPN multihop no-nexthop-change
user@MX22# set protocols bgp group MX21-EVPN local-address 198.51.100.22
user@MX22# set protocols bgp group MX21-EVPN peer-as 800
user@MX22# set protocols bgp group MX21-EVPN local-as 900
user@MX22# set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family
evpn signaling
```

5. Configure integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

   a. The following is the IRB gateway configuration for the VLAN-1 on Host (which is the host part of VLAN-1):

```
[edit]
```

```
user@MX22# set interfaces irb unit 1 proxy-macip-advertisement
user@MX22# set interfaces irb unit 1 virtual-gateway-esi
00:22:aa:aa:aa:aa:aa:aa:aa:aa
user@MX22# set interfaces irb unit 1 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 1 family inet address 10.11.1.15/24
virtual-gateway-address 10.11.1.11
```

   b. The following is the IRB gateway configuration for the VLAN-2 on Host (which is the host part of VLAN-2):

```
[edit]
```

```
user@MX22# set interfaces irb unit 2 proxy-macip-advertisement
user@MX22# set interfaces irb unit 2 virtual-gateway-esi
00:22:bb:bb:bb:bb:bb:bb:bb:bb
user@MX22# set interfaces irb unit 2 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 2 family inet address 10.12.1.15/24
virtual-gateway-address 10.12.1.11
```

   c. The following is the IRB gateway configuration for the VLAN-3 on Host (which is the host part of VLAN-3):

```
[edit]
```

```
user@MX22# set interfaces irb unit 3 proxy-macip-advertisement
user@MX22# set interfaces irb unit 3 virtual-gateway-esi
00:22:cc:cc:cc:cc:cc:cc:cc:cc
user@MX22# set interfaces irb unit 3 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 3 family inet address 10.13.1.15/24
virtual-gateway-address 10.13.1.11
```

   d. The following is the IRB gateway configuration for the VLAN-4 on Host (which is the host part of VLAN-4):

```
[edit]
```

```
user@MX22# set interfaces irb unit 4 proxy-macip-advertisement
user@MX22# set interfaces irb unit 4 virtual-gateway-esi
00:22:dd:dd:dd:dd:dd:dd:dd:dd
user@MX22# set interfaces irb unit 4 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 4 family inet address 10.14.1.15/24
virtual-gateway-address 10.14.1.11
```

e. The following is the IRB gateway configuration for the VLAN-5 on Host (which is the host part of VLAN-5):

```
[edit]
```

```
user@MX22# set interfaces irb unit 5 proxy-macip-advertisement
user@MX22# set interfaces irb unit 5 virtual-gateway-esi
00:22:ee:ee:ee:ee:ee:ee:ee:ee
user@MX22# set interfaces irb unit 5 virtual-gateway-esi all-active
user@MX22# set interfaces irb unit 5 family inet address 10.15.1.15/24
virtual-gateway-address 10.15.1.11
```

6. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@MX22# set protocols l2-learning traceoptions file MX22-L2ALD.log
user@MX22# set protocols l2-learning traceoptions file size 10m
user@MX22# set protocols l2-learning traceoptions level all
user@MX22# set protocols l2-learning traceoptions flag all
```

7. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@MX22# set policy-options policy-statement LO from protocol direct
user@MX22# set policy-options policy-statement LO from route-filter
198.51.100.22/32 exact
user@MX22# set policy-options policy-statement LO then accept
user@MX22# set policy-options policy-statement TEST1 term 1 from protocol
```

```
bgp
user@MX22# set policy-options policy-statement TEST1 term 1 from external
user@MX22# set policy-options policy-statement TEST1 term 1 then reject
```

8. Configure community policy options.

```
[edit]
```

```
user@MX22# set policy-options community NO-EXPORT members no-advertise
user@MX22# set policy-options community NO-EXPORT members no-export
user@MX22# set policy-options community NO-EXPORT members no-export-subconfed
```

9. Apply load balance.

```
[edit]
```

```
user@MX22# set policy-options policy-statement TEST then community add
NO-EXPORT
user@MX22# set policy-options policy-statement evpn-pplb from protocol evpn
user@MX22# set policy-options policy-statement evpn-pplb then load-balance
per-packet
```

10. Configure an ESI value on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC2 network.

```
[edit]
```

```
user@MX22# set interfaces lt-1/0/0 esi 00:33:33:33:33:33:33:33:33:33
```

11. Configure active-active multihoming on the logical tunnel interface by including the all-active statement.

```
[edit]
```

```
user@MX22# set interfaces lt-1/0/0 esi all-active
```

12. Configure a pair of logical tunnel (lt-) interface on the MX22 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the

MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
[edit]
```

```
user@MX22# set interfaces lt-1/0/0 unit 0 peer-unit 1
user@MX22# set interfaces lt-1/0/0 unit 0 family bridge interface-mode trunk
user@MX22# set interfaces lt-1/0/0 unit 0 family bridge vlan-id-list 1-5
user@MX22# set interfaces lt-1/0/0 unit 1 peer-unit 0
user@MX22# set interfaces lt-1/0/0 unit 1 family bridge interface-mode trunk
user@MX22# set interfaces lt-1/0/0 unit 1 family bridge vlan-id-list 1-5
```

13. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@MX22# set interfaces lo0 unit 88 family inet address 198.51.100.22/32
user@MX22# set routing-options router-id 198.51.100.22
user@MX22# set routing-options autonomous-system 300
```

14. Configure load balancing on MX22.

```
[edit]
```

```
user@MX22# set routing-options forwarding-table export evpn-pplb
```

15. Enable RSVP, MPLS, BGP, and OSPF protocols on the core interfaces. Create MPLS LSPs and specify the address of the other gateway and WAN edge routers (MX11, MX12, P, MX21).

```
[edit]
```

```
user@MX22# set protocols rsvp interface all
user@MX22# set protocols rsvp interface fxp0.0 disable
user@MX22# set protocols mpls label-switched-path MX22-TO-MX11 to 192.0.2.21
user@MX22# set protocols mpls label-switched-path MX22-TO-MX12 to 192.0.2.22
user@MX22# set protocols mpls label-switched-path MX22-TO-P to 203.0.113.1
user@MX22# set protocols mpls label-switched-path MX22-TO-MX21 to
198.51.100.21
```

```
user@MX22# set protocols mpls interface all
user@MX22# set protocols mpls interface fxp0.0 disable
user@MX22# set protocols bgp local-address 198.51.100.22
user@MX22# set protocols bgp export TEST
user@MX22# set protocols bgp local-as 300
user@MX22# set protocols bgp group INT type internal
user@MX22# set protocols bgp group INT family evpn signaling
user@MX22# set protocols bgp group INT export TEST1
user@MX22# set protocols bgp group INT neighbor 203.0.113.1
user@MX22# set protocols ospf traffic-engineering
user@MX22# set protocols ospf area 0.0.0.0 interface ge-1/0/2.0
user@MX22# set protocols ospf area 0.0.0.0 interface lo0.88 passive
```

16. Configure EVPN-based MPLS routing instances on the MX22 router for each virtual network. Define the route distinguisher (used to identify and advertise EVPN-MPLS routes) and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure a bridge domain for each virtual router that maps VLAN IDs.

```
[edit]
```

```
user@MX22# set routing-instances EVPN-MPLS-1 instance-type virtual-switch
user@MX22# set routing-instances EVPN-MPLS-1 interface lt-1/0/0.0
user@MX22# set routing-instances EVPN-MPLS-1 route-distinguisher
198.51.100.22:100
user@MX22# set routing-instances EVPN-MPLS-1 vrf-target target:1:2
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file MX22-EVPN-MPLS-1.log
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
file size 10m
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn traceoptions
flag all
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn extended-vlan-list
 1-5
user@MX22# set routing-instances EVPN-MPLS-1 protocols evpn default-gateway
 no-gateway-community
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 domain-type
 bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-1 vlan-id 1
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 domain-type
 bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-2 vlan-id 2
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 domain-type
 bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-3 vlan-id 3
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 domain-type
 bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-4 vlan-id 4
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 domain-type
```

```
 bridge
user@MX22# set routing-instances EVPN-MPLS-1 bridge-domains BD-5 vlan-id 5
```

17. Configure EVPN-VXLAN routing instances on the MX22 router for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@MX22# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.88
user@MX22# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@MX22# set routing-instances EVPN-VXLAN-1 interface lt-1/0/0.1
user@MX22# set routing-instances EVPN-VXLAN-1 route-distinguisher
198.51.100.22:1
user@MX22# set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file MX22-EVPN-VXLAN-1.log
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
vxlan
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn extended-vni-list
 1-5
user@MX22# set routing-instances EVPN-VXLAN-1 protocols evpn default-gateway
 no-gateway-community
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id 1
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1
routing-interface irb.1
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id 2
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2
routing-interface irb.2
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id 3
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3
routing-interface irb.3
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
```

```
 bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id 4
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4
routing-interface irb.4
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id 5
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5
routing-interface irb.5
user@MX22# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
user@MX22# set routing-instances VRF instance-type vrf
user@MX22# set routing-instances VRF interface irb.1
user@MX22# set routing-instances VRF interface irb.2
user@MX22# set routing-instances VRF interface irb.3
user@MX22# set routing-instances VRF interface irb.4
user@MX22# set routing-instances VRF interface irb.5
user@MX22# set routing-instances VRF route-distinguisher 1:1
user@MX22# set routing-instances VRF vrf-target target:10:10
```

Configuring ToR21

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router as ToR21:

1. Set the system hostname.

```
[edit]
```

```
user@ToR21# set system host-name ToR21
```

2. Configure the interfaces and bridge domains on the CE4 device to enable Layer 2 connectivity.

```
[edit]
```

```
user@ce4# set logical-systems CE-4 interfaces ge-1/0/9 unit 0 description
"CONNECTED TO Host 4"
user@ce4# set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge
 interface-mode trunk
user@ce4# set logical-systems CE-4 interfaces ge-1/0/9 unit 0 family bridge
 vlan-id-list 1-5
```

```
user@ce4# set logical-systems CE-4 interfaces ge-1/1/6 unit 0 description
"CONNECTED TO ToR21"
user@ce4# set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge
 interface-mode trunk
user@ce4# set logical-systems CE-4 interfaces ge-1/1/6 unit 0 family bridge
 vlan-id-list 1-5
user@ce4# set logical-systems CE-4 bridge-domains BD-1 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-1 vlan-id 1
user@ce4# set logical-systems CE-4 bridge-domains BD-2 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-2 vlan-id 2
user@ce4# set logical-systems CE-4 bridge-domains BD-3 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-3 vlan-id 3
user@ce4# set logical-systems CE-4 bridge-domains BD-4 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-4 vlan-id 4
user@ce4# set logical-systems CE-4 bridge-domains BD-5 domain-type bridge
user@ce4# set logical-systems CE-4 bridge-domains BD-5 vlan-id 5
```

3. Configure trace options for the interfaces to enable trace logs.

```
[edit]
```

```
user@ce4# set interfaces traceoptions file R6-DCD.log
user@ce4# set interfaces traceoptions file size 10m
user@ce4# set interfaces traceoptions flag all
```

4. Set the number of aggregated Ethernet interfaces.

```
[edit]
```

```
user@ToR21# set chassis aggregated-devices ethernet device-count 1
```

5. Configure the interfaces on the ToR21 device to connect to the MX22, CE-5, CE-4, ToR22, and MX21 devices to enable underlay connectivity.

```
[edit]
```

```
user@ToR21# set interfaces xe-0/0/0 unit 0 description "CONNECTED TO MX22"
user@ToR21# set interfaces xe-0/0/0 unit 0 family inet address 192.168.10.2/24
user@ToR21# set interfaces ge-1/0/0 description "CONNECTED TO CE-5"
user@ToR21# set interfaces ge-1/0/0 gigether-options 802.3ad ae0
user@ToR21# set interfaces ge-1/0/1 unit 0 description "CONNECTED TO MX21"
user@ToR21# set interfaces ge-1/0/1 unit 0 family inet address
192.168.101.1/24
user@ToR21# set interfaces ge-1/0/6 unit 0 description "CONNECTED TO CE-4"
```

```
user@ToR21# set interfaces ge-1/0/6 unit 0 family bridge interface-mode trunk
user@ToR21# set interfaces ge-1/0/6 unit 0 family bridge vlan-id-list 1-5
user@ToR21# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR22"
user@ToR21# set interfaces ge-1/1/3 unit 0 family inet address 192.168.12.1/24
```

6. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-5 end host device. The ESI value is globally unique across the entire EVPN domain. The all-active configuration enables ToR21 and ToR22 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```
[edit]
```

```
user@ToR21# set interfaces ae0 esi 00:44:44:44:44:44:44:44:44:44
user@ToR21# set interfaces ae0 esi all-active
user@ToR21# set interfaces ae0 aggregated-ether-options lacp active
user@ToR21# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@ToR21# set interfaces ae0 aggregated-ether-options lacp system-id
22:22:22:22:22:22
user@ToR21# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR21# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
```

7. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@ToR21# set interfaces lo0 unit 90 family inet address 198.51.100.11/32
user@ToR21# set routing-options router-id 198.51.100.11
user@ToR21# set routing-options autonomous-system 600
```

8. Configure load balancing on ToR21.

```
[edit]
```

```
user@ToR21# set routing-options forwarding-table export evpn-pplb
```

9. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR21 and ToR22) and gateway routers (MX21 and MX22).

```
[edit]
```

```
user@ToR21# set protocols bgp export TEST
user@ToR21# set protocols bgp local-as 600
user@ToR21# set protocols bgp group MX21 type external
user@ToR21# set protocols bgp group MX21 local-address 192.168.9.2
user@ToR21# set protocols bgp group MX21 export LO
user@ToR21# set protocols bgp group MX21 export TEST
user@ToR21# set protocols bgp group MX21 peer-as 800
user@ToR21# set protocols bgp group MX21 local-as 600
user@ToR21# set protocols bgp group MX21 neighbor 192.168.9.1 family inet
unicast
user@ToR21# set protocols bgp group MX22 type external
user@ToR21# set protocols bgp group MX22 local-address 10.102.2.1
user@ToR21# set protocols bgp group MX22 export LO
user@ToR21# set protocols bgp group MX22 export TEST
user@ToR21# set protocols bgp group MX22 peer-as 900
user@ToR21# set protocols bgp group MX22 local-as 600
user@ToR21# set protocols bgp group MX22 neighbor 192.168.10.1 family inet
unicast
user@ToR21# set protocols bgp group ToR22 type external
user@ToR21# set protocols bgp group ToR22 local-address 10.105.5.1
user@ToR21# set protocols bgp group ToR22 export LO
user@ToR21# set protocols bgp group ToR22 export TEST
user@ToR21# set protocols bgp group ToR22 peer-as 700
user@ToR21# set protocols bgp group ToR22 local-as 600
user@ToR21# set protocols bgp group ToR22 neighbor 192.168.12.2 family inet
 unicast
```

10. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR21 and ToR22) and gateway routers (MX21 and MX22) and set EVPN as the signaling protocol.

    a. Configure a MP-EBGP overlay to connect between ToR21 and MX21 using EVPN signaling.

    ```
    [edit]
    ```

    ```
    user@ToR21# set protocols bgp group MX21-EVPN type external
    user@ToR21# set protocols bgp group MX21-EVPN multihop ttl 2
    user@ToR21# set protocols bgp group MX21-EVPN multihop no-nexthop-change
    user@ToR21# set protocols bgp group MX21-EVPN local-address 198.51.100.11
    user@ToR21# set protocols bgp group MX21-EVPN peer-as 800
    user@ToR21# set protocols bgp group MX21-EVPN local-as 600
    user@ToR21# set protocols bgp group MX21-EVPN neighbor 198.51.100.21 family
     evpn signaling
    ```

    b. Configure a MP-EBGP overlay to connect between ToR21 and MX22 using EVPN signaling.

    ```
    [edit]
    ```

```
user@ToR21# set protocols bgp group MX22-EVPN type external
user@ToR21# set protocols bgp group MX22-EVPN multihop ttl 2
user@ToR21# set protocols bgp group MX22-EVPN multihop no-nexthop-change
user@ToR21# set protocols bgp group MX22-EVPN local-address 198.51.100.11
user@ToR21# set protocols bgp group MX22-EVPN peer-as 900
user@ToR21# set protocols bgp group MX22-EVPN local-as 600
user@ToR21# set protocols bgp group MX22-EVPN neighbor 198.51.100.22 family
 evpn signaling
```

c. Configure a MP-EBGP overlay to connect between ToR21 and ToR22 using EVPN signaling.

```
[edit]
```

```
user@ToR21# set protocols bgp group ToR22-EVPN type external
user@ToR21# set protocols bgp group ToR22-EVPN multihop ttl 2
user@ToR21# set protocols bgp group ToR22-EVPN multihop no-nexthop-change
user@ToR21# set protocols bgp group ToR22-EVPN local-address 198.51.100.11
user@ToR21# set protocols bgp group ToR22-EVPN peer-as 700
user@ToR21# set protocols bgp group ToR22-EVPN local-as 600
user@ToR21# set protocols bgp group ToR22-EVPN neighbor 198.51.100.12
family evpn signaling
```

11. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@ToR21# set protocols l2-learning traceoptions file TOR21-L2ALD.log
user@ToR21# set protocols l2-learning traceoptions file size 10m
user@ToR21# set protocols l2-learning traceoptions level all
user@ToR21# set protocols l2-learning traceoptions flag all
```

12. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@ToR21# set policy-options policy-statement LO term 1 from protocol
direct
user@ToR21# set policy-options policy-statement LO term 1 from route-filter
```

```
 198.51.100.11/32 exact
user@ToR21# set policy-options policy-statement LO term 1 then accept
```

13. Configure community policy options.

```
[edit]
```

```
user@ToR21# set policy-options community NO-EXPORT members no-advertise
user@ToR21# set policy-options community NO-EXPORT members no-export
user@ToR21# set policy-options community NO-EXPORT members no-export-subconfed
```

14. Apply load balance.

```
[edit]
```

```
user@ToR21# set policy-options policy-statement TEST then community add
NO-EXPORT
user@ToR21# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR21# set policy-options policy-statement evpn-pplb then load-balance
 per-packet
```

15. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@ToR21# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.90
user@ToR21# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR21# set routing-instances EVPN-VXLAN-1 interface ge-1/0/6.0
user@ToR21# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR21# set routing-instances EVPN-VXLAN-1 route-distinguisher
198.51.100.11:1
user@ToR21# set routing-instances EVPN-VXLAN-1 vrf-target target:1:3
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file TOR21-EVPN-VXLAN-1.log
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
```

```
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
 vxlan
user@ToR21# set routing-instances EVPN-VXLAN-1 protocols evpn
extended-vni-list 1-5
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id
1
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id
2
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id
3
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
 bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id
4
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id
5
user@ToR21# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
```

Configuring ToR22

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Configure the MX router ToR22:

1. Set the system hostname.

```
[edit]
```

```
user@ToR22# set system host-name ToR22
```

2. Configure the interfaces and bridge domains on the CE-5 device to enable Layer 2 connectivity.

```
[edit]
```

```
user@ce5# set logical-systems CE-5 interfaces ge-1/0/9 unit 0 description
"CONNECTED TO Host 5"
user@ce5# set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge
 interface-mode trunk
user@ce5# set logical-systems CE-5 interfaces ge-1/0/9 unit 0 family bridge
 vlan-id-list 1-5
user@ce5# set logical-systems CE-5 interfaces ae1 unit 0 description
"CONNECTED TO ToR21"
user@ce5# set logical-systems CE-5 interfaces ae1 unit 0 family bridge
interface-mode trunk
user@ce5# set logical-systems CE-5 interfaces ae1 unit 0 family bridge
vlan-id-list 1-5
user@ce5# set logical-systems CE-5 bridge-domains BD-1 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-1 vlan-id 1
user@ce5# set logical-systems CE-5 bridge-domains BD-2 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-2 vlan-id 2
user@ce5# set logical-systems CE-5 bridge-domains BD-3 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-3 vlan-id 3
user@ce5# set logical-systems CE-5 bridge-domains BD-4 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-4 vlan-id 4
user@ce5# set logical-systems CE-5 bridge-domains BD-5 domain-type bridge
user@ce5# set logical-systems CE-5 bridge-domains BD-5 vlan-id 5
```

3. Configure the interfaces and bridge domains on the CE-3 device to enable Layer 2 connectivity.

```
[edit]
```

```
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 description
"CONNECTED TO ToR12"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge
 interface-mode trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/7 unit 0 family bridge
 vlan-id-list 1-5
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 description
"CONNECTED TO Host 3"
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge
 interface-mode trunk
user@ce3# set logical-systems CE-3 interfaces ge-1/1/9 unit 0 family bridge
 vlan-id-list 1-5
user@ce3# set logical-systems CE-3 bridge-domains BD-1 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-1 vlan-id 1
user@ce3# set logical-systems CE-3 bridge-domains BD-2 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-2 vlan-id 2
user@ce3# set logical-systems CE-3 bridge-domains BD-3 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-3 vlan-id 3
user@ce3# set logical-systems CE-3 bridge-domains BD-4 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-4 vlan-id 4
```

```
user@ce3# set logical-systems CE-3 bridge-domains BD-5 domain-type bridge
user@ce3# set logical-systems CE-3 bridge-domains BD-5 vlan-id 5
```

4. Configure trace options for the interfaces to enable trace logs.

```
[edit]
```

```
user@ce3# set interfaces traceoptions file R1-DCD.log
user@ce3# set interfaces traceoptions file size 10m
user@ce3# set interfaces traceoptions flag all
```

5. Set the number of aggregated Ethernet interfaces.

```
[edit]
```

```
user@ToR12# set chassis aggregated-devices ethernet device-count 2
```

6. Configure the interfaces on the ToR12 device to connect to the MX12, CE-2, CE-3, ToR11, and MX11 devices to enable underlay connectivity.

```
[edit]
```

```
user@ToR12# set interfaces ge-1/0/0 unit 0 description "CONNECTED TO  MX11"
user@ToR12# set interfaces ge-1/0/0 unit 0 family inet address 192.168.6.1/24
user@ToR12# set interfaces ge-1/0/4 unit 0 description "CONNECTED TO MX12"
user@ToR12# set interfaces ge-1/0/4 unit 0 family inet address 192.168.5.1/24
user@ToR12# set interfaces ge-1/0/6 description "CONNECTED TO CE-1"
user@ToR12# set interfaces ge-1/0/6 gigether-options 802.3ad ae0
user@ToR12# set interfaces ge-1/0/7 unit 0 description "CONNECTED TO CE-3"
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ge-1/0/7 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ge-1/1/0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/0 gigether-options 802.3ad ae1
user@ToR12# set interfaces ge-1/1/3 unit 0 description "CONNECTED TO ToR11"
user@ToR12# set interfaces ge-1/1/3 unit 0 family inet address 192.168.2.2/24
user@ToR12# set interfaces ge-1/1/6 description "CONNECTED TO ToR12"
user@ToR12# set interfaces ge-1/1/6 gigether-options 802.3ad ae1
```

7. Configure a Link Aggregation Control Protocol (LACP)-enabled link aggregation group (LAG) interface towards the CE-1 end host device. The ESI value is globally unique across the entire EVPN domain. The all-active configuration enables ToR11

and ToR12 to forward traffic to, and from the CE devices, such that all CE links are actively used.

```
[edit]
```

```
user@ToR12# set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
user@ToR12# set interfaces ae0 esi all-active
user@ToR12# set interfaces ae0 aggregated-ether-options lacp system-id
11:11:11:11:11:11
user@ToR12# set interfaces ae0 unit 0 family bridge interface-mode trunk
user@ToR12# set interfaces ae0 unit 0 family bridge vlan-id-list 1-5
user@ToR12# set interfaces ae1 aggregated-ether-options lacp active
user@ToR12# set interfaces ae1 aggregated-ether-options lacp periodic fast
```

8. Configure the loopback interface address and routing options.

```
[edit]
```

```
user@ToR12# set interfaces lo0 unit 82 family inet address 192.0.2.12/32
user@ToR12# set routing-options router-id 192.0.2.12
user@ToR12# set routing-options autonomous-system 200
```

9. Configure load balancing on ToR12.

```
[edit]
```

```
user@ToR12# set routing-options forwarding-table export evpn-pplb
```

10. Configure a multiprotocol external BGP (MP-EBGP) underlay connectivity between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12).

```
[edit]
```

```
user@ToR12# set protocols bgp local-as 200
user@ToR12# set protocols bgp group MX11 type external
user@ToR12# set protocols bgp group MX11 local-address 192.168.6.1
user@ToR12# set protocols bgp group MX11 export LO
user@ToR12# set protocols bgp group MX11 export TEST
user@ToR12# set protocols bgp group MX11 peer-as 400
user@ToR12# set protocols bgp group MX11 local-as 200
user@ToR12# set protocols bgp group MX11 neighbor 192.168.6.2 family inet
```

```
unicast
user@ToR12# set protocols bgp group MX12 type external
user@ToR12# set protocols bgp group MX12 local-address 192.168.5.1
user@ToR12# set protocols bgp group MX12 export LO
user@ToR12# set protocols bgp group MX12 export TEST
user@ToR12# set protocols bgp group MX12 peer-as 500
user@ToR12# set protocols bgp group MX12 local-as 200
user@ToR12# set protocols bgp group MX12 neighbor 192.168.5.2 family inet
unicast
user@ToR12# set protocols bgp group ToR11 type external
user@ToR12# set protocols bgp group ToR11 local-address 192.168.2.2
user@ToR12# set protocols bgp group ToR11 export LO
user@ToR12# set protocols bgp group ToR11 export TEST
user@ToR12# set protocols bgp group ToR11 peer-as 100
user@ToR12# set protocols bgp group ToR11 local-as 200
user@ToR12# set protocols bgp group ToR11 neighbor 192.168.2.1 family inet
unicast
```

11. Configure a multiprotocol external BGP (MP-EBGP) overlay between the ToR (ToR12 and ToR11) and gateway routers (MX11 and MX12) and set EVPN as the signaling protocol.

    a. Configure a MP-EBGP overlay to connect between ToR12 and MX11 using EVPN signaling.

    ```
    [edit]
    ```

    ```
    user@ToR12# set protocols bgp group MX11-EVPN type external
    user@ToR12# set protocols bgp group MX11-EVPN multihop ttl 2
    user@ToR12# set protocols bgp group MX11-EVPN multihop no-nexthop-change
    user@ToR12# set protocols bgp group MX11-EVPN local-address 192.0.2.12
    user@ToR12# set protocols bgp group MX11-EVPN export TEST
    user@ToR12# set protocols bgp group MX11-EVPN peer-as 400
    user@ToR12# set protocols bgp group MX11-EVPN local-as 200
    user@ToR12# set protocols bgp group MX11-EVPN neighbor 192.0.2.21 family
     evpn signaling
    ```

    b. Configure a MP-EBGP overlay to connect between ToR12 and MX12 using EVPN signaling.

    ```
    [edit]
    ```

    ```
    user@ToR12# set protocols bgp group MX12-EVPN type external
    user@ToR12# set protocols bgp group MX12-EVPN multihop ttl 2
    user@ToR12# set protocols bgp group MX12-EVPN multihop no-nexthop-change
    user@ToR12# set protocols bgp group MX12-EVPN local-address 192.0.2.12
    user@ToR12# set protocols bgp group MX12-EVPN export TEST
    user@ToR12# set protocols bgp group MX12-EVPN peer-as 500
    ```

```
user@ToR12# set protocols bgp group MX12-EVPN local-as 200
user@ToR12# set protocols bgp group MX12-EVPN neighbor 192.0.2.22 family
 evpn signaling
```

c. Configure a MP-EBGP overlay to connect between ToR12 and ToR11 using EVPN signaling.

```
[edit]
```

```
user@ToR12# set protocols bgp group ToR11-EVPN type external
user@ToR12# set protocols bgp group ToR11-EVPN multihop ttl 2
user@ToR12# set protocols bgp group ToR11-EVPN multihop no-nexthop-change
user@ToR12# set protocols bgp group ToR11-EVPN local-address 192.0.2.12
user@ToR12# set protocols bgp group ToR11-EVPN export TEST
user@ToR12# set protocols bgp group ToR11-EVPN peer-as 100
user@ToR12# set protocols bgp group ToR11-EVPN local-as 200
user@ToR12# set protocols bgp group ToR11-EVPN neighbor 192.0.2.11 family
 evpn signaling
user@ToR12# set protocols bgp group ToR12-EVPN export TEST
```

12. Configure trace operations to track all Layer 2 address learning and forwarding properties.

```
[edit]
```

```
user@ToR12# set protocols l2-learning traceoptions file TOR12-L2ALD.log
user@ToR12# set protocols l2-learning traceoptions file size 10m
user@ToR12# set protocols l2-learning traceoptions level all
user@ToR12# set protocols l2-learning traceoptions flag all
```

13. Configure routing policies to accept the direct loopback address route and redirect it into BGP.

```
[edit]
```

```
user@ToR12# set policy-options policy-statement LO term 1 from protocol
direct
user@ToR12# set policy-options policy-statement LO term 1 from route-filter
 192.0.2.12/32 exact
user@ToR12# set policy-options policy-statement LO term 1 then accept
```

14. Configure community policy options.

```
[edit]
```

```
user@ToR12# set policy-options community NO-EXPORT members no-advertise
user@ToR12# set policy-options community NO-EXPORT members no-export
user@ToR12# set policy-options community NO-EXPORT members no-export-subconfed
```

15. Apply load balance.

```
[edit]
```

```
user@ToR12# set policy-options policy-statement TEST then community add
NO-EXPORT
user@ToR12# set policy-options policy-statement evpn-pplb from protocol evpn
user@ToR12# set policy-options policy-statement evpn-pplb then load-balance
 per-packet
```

16. Configure EVPN routing instances for each virtual network. Define the VTEP source interface, route distinguisher (used to identify and advertise EVPN routes), and vrf-target (exports and tags all routes for that local VRF using the defined route target). Configure the EVPN protocol, encapsulation method, VNI list, and BUM traffic forwarding method. Finally, configure a bridge domain for each virtual router that maps VNIDs to VLAN IDs, and identify the BUM forwarding method.

```
[edit]
```

```
user@ToR12# set routing-instances EVPN-VXLAN-1 vtep-source-interface lo0.82
user@ToR12# set routing-instances EVPN-VXLAN-1 instance-type virtual-switch
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ge-1/0/7.0
user@ToR12# set routing-instances EVPN-VXLAN-1 interface ae0.0
user@ToR12# set routing-instances EVPN-VXLAN-1 route-distinguisher
192.0.2.12:1
user@ToR12# set routing-instances EVPN-VXLAN-1 vrf-target target:1:1
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file TOR12-EVPN-VXLAN-1.log
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
file size 10m
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn traceoptions
flag all
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn encapsulation
 vxlan
user@ToR12# set routing-instances EVPN-VXLAN-1 protocols evpn
extended-vni-list 1-5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vlan-id
```

```
1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-1 vxlan vni
 1
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vlan-id
2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-2 vxlan vni
 2
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vlan-id
3
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-3 vxlan vni
 3
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vlan-id
4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-4 vxlan vni
 4
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 domain-type
 bridge
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vlan-id
5
user@ToR12# set routing-instances EVPN-VXLAN-1 bridge-domains BD-5 vxlan vni
 5
```

Verification

After you configure both the underlay and EVPN overlay we recommend that you verify that the configurations work as you intended.

- Verifying ToR11 Configuration on page 842
- Verifying ToR12 Configuration on page 848
- Verifying Data Center Gateway and WAN Edge 1 Router (MX11) Configuration on page 856
- Verifying Data Center Gateway and WAN Edge 2 Router (MX12) Configuration on page 865
- Verifying Data Center Gateway and WAN Edge 3 Router (MX21) Configuration on page 874
- Verifying Data Center Gateway and WAN Edge 4 Router (MX22) Configuration on page 883
- Verifying ToR21 Configuration on page 892
- Verifying ToR22 Configuration on page 898

Verifying ToR11 Configuration

Purpose Verify that ToR11 is properly configured.

Action Verify that the logical system interfaces and bridge domains on the CE2 device are properly configured to enable Layer 2 connectivity.

```
user@ce2> show configuration logical-systems
```

```
CE-2 {
    interfaces {
        ge-1/0/9 {
            unit 0 {
                description "CONNECTED TO Host 2";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
        ge-1/1/6 {
            unit 0 {
                description "CONNECTED TO ToR11";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }

    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;

        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;

        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;

        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;

        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;

        }
    }
}
```

Verify that the interfaces and trace options on ToR11 are configured properly to enable

underlay connectivity to other ToR and gateway and WAN edge devices.

```
user@ToR11>show configuration interfaces
```

```
traceoptions {
    file RO-DCD.log size 10m;
    flag all;
}
ge-1/0/5 {
    unit 0 {
        description "CONNECTED TO MX12";
        family inet {
            address 192.168.4.1/24;
        }
    }
}
ge-1/0/6 {
    unit 0 {
        description "CONNECTED TO CE-2";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/1/0 {
    description "CONNECTED TO CE-1";
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/1 {
    unit 0 {
        description "CONNECTED TO MX11";
        family inet {
            address 192.168.3.1/24;
        }
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR12";
        family inet {
            address 192.168.2.1/24;
        }
    }
}
ae0 {
    esi {
        00:11:11:11:11:11:11:11:11:11;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 11:11:11:11:11:11;
        }
    }
```

```
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
lo0 {
    unit 81 {
        family inet {
            address 192.0.2.11/32;
        }
    }
}
```

Verify that the routing and load balancing options are properly configured.

```
user@ToR11> show configuration routing-options
```

```
router-id 192.0.2.11;
autonomous-system 100;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```
user@ToR11> show configuration protocols
```

```
bgp {
    local-as 100;
    group MX11 {
        type external;
        local-address 192.168.3.1;
        export [ LO TEST ];
        peer-as 400;
        neighbor 192.168.3.2 {
            family inet {
                unicast;
            }
        }
    }
    group MX12 {
        type external;
        local-address 192.168.4.1;
        export [ LO TEST ];
        peer-as 500;
        neighbor 192.168.4.2 {
            family inet {
                unicast;
            }
        }
```

```
}
group ToR12 {
    type external;
    local-address 192.168.2.1;
    export [ LO TEST ];
    peer-as 200;
    local-as 100;
    neighbor 192.168.2.2 {
        family inet {
            unicast;
        }
    }
}
group MX11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.11;
    export TEST;
    peer-as 400;
    local-as 100;
    neighbor 192.0.2.21 {
        family evpn {
            signaling;
        }
    }
}
group MX12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.11;
    export TEST;
    peer-as 500;
    local-as 100;
    neighbor 192.0.2.22 {
        family evpn {
            signaling;
        }
    }
}
group ToR12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.11;
    export TEST;
    peer-as 200;
    local-as 100;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
```

```
        }
    }
}
l2-learning {
    traceoptions {
        file TOR11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@ToR11> show configuration policy-options
```

```
policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.11/32 exact;
        }
        then accept;
    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly configured.

```
user@ToR11> show configuration routing-instances
```

```
EVPN-VXLAN-1 {
    vtep-source-interface lo0.81;
    instance-type virtual-switch;
    interface ge-1/0/6.0;
    interface ae0.0;
    route-distinguisher 192.0.2.11:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
```

```
                file TOR11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
            vxlan {
                vni 5;
            }
        }
    }
}
```

Verifying ToR12 Configuration

Purpose Verify that ToR12 is properly configured.

Action Verify that the logical system interfaces and bridge domains on the CE1 and CE3 devices are properly configured to enable Layer 2 connectivity.

```
user@ce1> show configuration logical-systems
```

```
CE-1 {
    interfaces {
        ge-1/0/9 {
            unit 0 {
                description "CONNECTED TO Host 1";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
        ae1 {
            unit 0 {
                description "CONNECTED TO ToR12";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }

    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;

        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;

        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;

        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;

        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;

        }
    }
}
```

```
user@ce3> show configuration logical-systems
```

```
CE-3 {
    interfaces {
        ge-1/1/7 {
            unit 0 {
                description "CONNECTED TO ToR12";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
        ge-1/1/9 {
            unit 0 {
                description "CONNECTED TO Host 3";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }

    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;

        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;

        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;

        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;

        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;

        }
    }
}
```

Verify that the interfaces and trace options on ToR12 are configured properly to enable underlay connectivity to other ToR and gateway and WAN edge devices.

```
user@ToR12>show configuration interfaces
```

```
traceoptions {
    file R1-DCD.log size 10m;
    flag all;
}
ge-1/0/0 {
    unit 0 {
        description "CONNECTED TO  MX11";
        family inet {
            address 192.168.6.1/24;
        }
    }
}
ge-1/0/4 {
    unit 0 {
        description "CONNECTED TO MX12";
        family inet {
            address 192.168.5.1/24;
        }
    }
}
ge-1/0/6 {
    description "CONNECTED TO CE-1";
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/0/7 {
    unit 0 {
        description "CONNECTED TO CE-3";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/1/0 {
    description "CONNECTED TO ToR11";
    gigether-options {
        802.3ad ae1;
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR11";
        family inet {
            address 192.168.2.2/24;
        }
    }
}
ge-1/1/6 {
    description "CONNECTED TO ToR12";
    gigether-options {
        802.3ad ae1;
    }
}
ae0 {
    esi {
        00:11:11:11:11:11:11:11:11:11;
```

```
        all-active;
    }
    aggregated-ether-options {
        lacp {
            system-id 11:11:11:11:11:11;
        }
    }
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ae1 {
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
        }
    }
}
lo0 {
    unit 82 {
        family inet {
            address 192.0.2.12/32;
        }
    }
}
```

Verify that the routing and load balancing options are properly configured.

```
user@ToR12> show configuration routing-options
```

```
router-id 192.0.2.12;
autonomous-system 200;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```
user@ToR12> show configuration protocols
```

```
bgp {
    local-as 200;
    group MX11 {
        type external;
        local-address 192.168.6.1;
        export [ LO TEST ];
        peer-as 400;
        local-as 200;
```

```
    neighbor 192.168.6.2 {
        family inet {
            unicast;
        }
    }
}
group MX12 {
    type external;
    local-address 192.168.5.1;
    export [ LO TEST ];
    peer-as 500;
    local-as 200;
    neighbor 192.168.5.2 {
        family inet {
            unicast;
        }
    }
}
group ToR11 {
    type external;
    local-address 192.168.2.2;
    export [ LO TEST ];
    peer-as 100;
    local-as 200;
    neighbor 192.168.2.1 {
        family inet {
            unicast;
        }
    }
}
group MX11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.12;
    export TEST;
    peer-as 400;
    local-as 200;
    neighbor 192.0.2.21 {
        family evpn {
            signaling;
        }
    }
}
group MX12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.12;
    export TEST;
    peer-as 500;
    local-as 200;
    neighbor 192.0.2.22 {
        family evpn {
            signaling;
```

```
            }
        }
    }
    group ToR11-EVPN {
        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 192.0.2.12;
        export TEST;
        peer-as 100;
        local-as 200;
        neighbor 192.0.2.11 {
            family evpn {
                signaling;
            }
        }
    }
    group ToR12-EVPN {
        export TEST;
    }
}
l2-learning {
    traceoptions {
        file TOR12-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@ToR12> show configuration policy-options
```

```
policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.12/32 exact;
        }
        then accept;
    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
```

```
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly configured.

```
user@ToR12> show configuration routing-instances
```

```
EVPN-VXLAN-1 {
    vtep-source-interface lo0.82;
    instance-type virtual-switch;
    interface ge-1/0/7.0;
    interface ae0.0;
    route-distinguisher 192.0.2.12:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file TOR12-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
```

```
            vxlan {
                vni 5;
            }
        }
    }
}
```

Verifying Data Center Gateway and WAN Edge 1 Router (MX11) Configuration

Purpose Verify that MX11 is properly configured.

Action Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
user@MX11> show configuration interfaces
```

```
traceoptions {
    file R2-DCD.log size 10m;
    flag all;
}
ge-5/1/0 {
    unit 0 {
        description "CONNECTED TO MX-21";
        family inet {
            address 192.168.7.1/24;
        }
    }
}
ge-5/1/1 {
    unit 0 {
        description "CONNECTED TO TOR-11";
        family inet {
            address 192.168.3.2/24;
        }
    }
}
ge-5/1/8 {
    unit 0 {
        description "CONNECTED TO TOR-12";
        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}
```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```
user@MX11> show configuration interfaces
```

```
irb {
    unit 1 {
        proxy-macip-advertisement;
```

```
        virtual-gateway-esi {
            00:11:aa:aa:aa:aa:aa:aa:aa:aa;
            all-active;
        }
        family inet {
            address 10.11.1.12/24 {
                virtual-gateway-address 10.11.1.10;
            }
        }
    }
    unit 2 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:bb:bb:bb:bb:bb:bb:bb:bb;
            all-active;
        }
        family inet {
            address 10.12.1.12/24 {
                virtual-gateway-address 10.12.1.10;
            }
        }
    }
    unit 3 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:cc:cc:cc:cc:cc:cc:cc:cc;
            all-active;
        }
        family inet {
            address 10.13.1.12/24 {
                virtual-gateway-address 10.13.1.10;
            }
        }
    }
    unit 4 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:dd:dd:dd:dd:dd:dd:dd:dd;
            all-active;
        }
        family inet {
            address 10.14.1.12/24 {
                virtual-gateway-address 10.14.1.10;
            }
        }
    }
    unit 5 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:ee:ee:ee:ee:ee:ee:ee:ee;
            all-active;
        }
        family inet {
            address 10.15.1.12/24 {
                virtual-gateway-address 10.15.1.10;
            }
        }
    }
}
```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    esi {
        00:22:22:22:22:22:22:22:22:22;
        all-active;
    }
}
```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    unit 0 {
        peer-unit 1;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
    unit 1 {
        peer-unit 0;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
```

Loopback interface address.

```
user@MX11> show configuration interfaces
```

```
lo0 {
    unit 84 {
        family inet {
            address 192.0.2.21/32;
        }
        family mpls;
    }
}
```

Verify that the routing options and load balancing are properly configured.

```
user@MX11> show configuration routing-options
```

```
router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```
user@MX11> show configuration protocols
```

```
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path MX11-TO-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-TO-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-TO-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-TO-MX22 {
        to 198.51.100.22;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {
            signaling;
        }
        export TEST;
        neighbor 203.0.113.1;
    }
    group TOR-11 {
        type external;
        local-address 192.168.3.2;
        import TEST;
        export [ TEST LO ];
        peer-as 100;
        local-as 400;
        neighbor 192.168.3.1 {
```

```
        family inet {
            unicast;
        }
    }
}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST LO ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}
group TOR-11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}
group TOR-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 200;
    local-as 400;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
```

```
        }
    }
    group MX-12-EVPN {
        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 192.0.2.21;
        export TEST;
        peer-as 500;
        local-as 400;
        neighbor 192.0.2.22 {
            family evpn {
                signaling;
            }
        }
    }
    group MX-11-EVPN {
        export TEST;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}
l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@MX11> show configuration policy-options
```

```
policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.21/32 exact;
        }
        then accept;
    }
    from {
        protocol direct;
        route-filter 192.0.2.21/32 exact;
    }
```

```
    then accept;
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```
user@MX11> show configuration routing-instances
```

```
EVPN-MPLS-1 {
    instance-type virtual-switch;
    interface lt-5/1/0.0;
    route-distinguisher 192.0.2.21:100;
    vrf-target target:1:2;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-MPLS-1.log size 10m;
                flag all;
            }
            extended-vlan-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
        }
```

```
    }
}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            routing-interface irb.1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            routing-interface irb.2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            routing-interface irb.3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            routing-interface irb.4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
            routing-interface irb.5;
            vxlan {
                vni 5;
            }
```

```
        }
    }
}
VRF {
    instance-type vrf;
    interface irb.1;
    interface irb.2;
    interface irb.3;
    interface irb.4;
    interface irb.5;
    route-distinguisher 1:1;
    vrf-target target:10:10;
}
```

Verifying Data Center Gateway and WAN Edge 2 Router (MX12) Configuration

Purpose Verify that MX12 is properly configured.

Action Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
user@MX11> show configuration interfaces
```

```
traceoptions {
    file R2-DCD.log size 10m;
    flag all;
}
ge-5/1/0 {
    unit 0 {
        description "CONNECTED TO MX-21";
        family inet {
            address 192.168.7.1/24;
        }
    }
}
ge-5/1/1 {
    unit 0 {
        description "CONNECTED TO TOR-11";
        family inet {
            address 192.168.3.2/24;
        }
    }
}
ge-5/1/8 {
    unit 0 {
        description "CONNECTED TO TOR-12";
        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}
```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```
user@MX11> show configuration interfaces
```

```
irb {
    unit 1 {
        proxy-macip-advertisement;
```

```
        virtual-gateway-esi {
            00:11:aa:aa:aa:aa:aa:aa:aa:aa;
            all-active;
        }
        family inet {
            address 10.11.1.12/24 {
                virtual-gateway-address 10.11.1.10;
            }
        }
    }
    unit 2 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:bb:bb:bb:bb:bb:bb:bb:bb;
            all-active;
        }
        family inet {
            address 10.12.1.12/24 {
                virtual-gateway-address 10.12.1.10;
            }
        }
    }
    unit 3 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:cc:cc:cc:cc:cc:cc:cc:cc;
            all-active;
        }
        family inet {
            address 10.13.1.12/24 {
                virtual-gateway-address 10.13.1.10;
            }
        }
    }
    unit 4 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:dd:dd:dd:dd:dd:dd:dd:dd;
            all-active;
        }
        family inet {
            address 10.14.1.12/24 {
                virtual-gateway-address 10.14.1.10;
            }
        }
    }
    unit 5 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:ee:ee:ee:ee:ee:ee:ee:ee;
            all-active;
        }
        family inet {
            address 10.15.1.12/24 {
                virtual-gateway-address 10.15.1.10;
            }
        }
    }
}
```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    esi {
        00:22:22:22:22:22:22:22:22:22;
        all-active;
    }
}
```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    unit 0 {
        peer-unit 1;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
    unit 1 {
        peer-unit 0;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
```

Loopback interface address.

```
user@MX11> show configuration interfaces
```

```
lo0 {
    unit 84 {
        family inet {
            address 192.0.2.21/32;
        }
        family mpls;
    }
}
```

Verify that the routing options and load balancing are properly configured.

```
user@MX11> show configuration routing-options
```

```
router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```
user@MX11> show configuration protocols
```

```
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path MX11-TO-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-TO-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-TO-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-TO-MX22 {
        to 198.51.100.22;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {
            signaling;
        }
        export TEST;
        neighbor 203.0.113.1;
    }
    group TOR-11 {
        type external;
        local-address 192.168.3.2;
        import TEST;
        export [ TEST LO ];
        peer-as 100;
        local-as 400;
        neighbor 192.168.3.1 {
```

```
        family inet {
            unicast;
        }
    }
}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST LO ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}
group TOR-11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}
group TOR-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 200;
    local-as 400;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
```

```
        }
    }
    group MX-12-EVPN {
        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 192.0.2.21;
        export TEST;
        peer-as 500;
        local-as 400;
        neighbor 192.0.2.22 {
            family evpn {
                signaling;
            }
        }
    }
    group MX-11-EVPN {
        export TEST;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}
l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@MX11> show configuration policy-options
```

```

policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.21/32 exact;
        }
        then accept;
    }
    from {
        protocol direct;
        route-filter 192.0.2.21/32 exact;
    }
```

```
    then accept;
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```
user@MX11> show configuration routing-instances
```

```

EVPN-MPLS-1 {
    instance-type virtual-switch;
    interface lt-5/1/0.0;
    route-distinguisher 192.0.2.21:100;
    vrf-target target:1:2;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-MPLS-1.log size 10m;
                flag all;
            }
            extended-vlan-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
        }
```

```
    }
}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            routing-interface irb.1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            routing-interface irb.2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            routing-interface irb.3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            routing-interface irb.4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
            routing-interface irb.5;
            vxlan {
                vni 5;
            }
```

```
        }
    }
}
VRF {
    instance-type vrf;
    interface irb.1;
    interface irb.2;
    interface irb.3;
    interface irb.4;
    interface irb.5;
    route-distinguisher 1:1;
    vrf-target target:10:10;
}
```

Verifying Data Center Gateway and WAN Edge 3 Router (MX21) Configuration

Purpose Verify that MX21 is properly configured.

Action Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
user@MX11> show configuration interfaces
```

```
traceoptions {
    file R2-DCD.log size 10m;
    flag all;
}
ge-5/1/0 {
    unit 0 {
        description "CONNECTED TO MX-21";
        family inet {
            address 192.168.7.1/24;
        }
    }
}
ge-5/1/1 {
    unit 0 {
        description "CONNECTED TO TOR-11";
        family inet {
            address 192.168.3.2/24;
        }
    }
}
ge-5/1/8 {
    unit 0 {
        description "CONNECTED TO TOR-12";
        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}
```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```
user@MX11> show configuration interfaces
```

```
irb {
    unit 1 {
        proxy-macip-advertisement;
```

```
        virtual-gateway-esi {
            00:11:aa:aa:aa:aa:aa:aa:aa:aa;
            all-active;
        }
        family inet {
            address 10.11.1.12/24 {
                virtual-gateway-address 10.11.1.10;
            }
        }
    }
    unit 2 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:bb:bb:bb:bb:bb:bb:bb:bb;
            all-active;
        }
        family inet {
            address 10.12.1.12/24 {
                virtual-gateway-address 10.12.1.10;
            }
        }
    }
    unit 3 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:cc:cc:cc:cc:cc:cc:cc:cc;
            all-active;
        }
        family inet {
            address 10.13.1.12/24 {
                virtual-gateway-address 10.13.1.10;
            }
        }
    }
    unit 4 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:dd:dd:dd:dd:dd:dd:dd:dd;
            all-active;
        }
        family inet {
            address 10.14.1.12/24 {
                virtual-gateway-address 10.14.1.10;
            }
        }
    }
    unit 5 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:ee:ee:ee:ee:ee:ee:ee:ee;
            all-active;
        }
        family inet {
            address 10.15.1.12/24 {
                virtual-gateway-address 10.15.1.10;
            }
        }
    }
}
```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    esi {
        00:22:22:22:22:22:22:22:22:22;
        all-active;
    }
}
```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    unit 0 {
        peer-unit 1;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
    unit 1 {
        peer-unit 0;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
```

Loopback interface address.

```
user@MX11> show configuration interfaces
```

```
lo0 {
    unit 84 {
        family inet {
            address 192.0.2.21/32;
        }
        family mpls;
    }
}
```

Verify that the routing options and load balancing are properly configured.

```
user@MX11> show configuration routing-options
```

```
router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```
user@MX11> show configuration protocols
```

```
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path MX11-TO-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-TO-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-TO-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-TO-MX22 {
        to 198.51.100.22;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {
            signaling;
        }
        export TEST;
        neighbor 203.0.113.1;
    }
    group TOR-11 {
        type external;
        local-address 192.168.3.2;
        import TEST;
        export [ TEST LO ];
        peer-as 100;
        local-as 400;
        neighbor 192.168.3.1 {
```

```
        family inet {
            unicast;
        }
    }
}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST LO ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}
group TOR-11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}
group TOR-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 200;
    local-as 400;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
```

```
        }
    }
    group MX-12-EVPN {
        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 192.0.2.21;
        export TEST;
        peer-as 500;
        local-as 400;
        neighbor 192.0.2.22 {
            family evpn {
                signaling;
            }
        }
    }
    group MX-11-EVPN {
        export TEST;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}
l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@MX11> show configuration policy-options
```

```
policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.21/32 exact;
        }
        then accept;
    }
    from {
        protocol direct;
        route-filter 192.0.2.21/32 exact;
    }
```

```
    then accept;
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```
user@MX11> show configuration routing-instances
```

```
EVPN-MPLS-1 {
    instance-type virtual-switch;
    interface lt-5/1/0.0;
    route-distinguisher 192.0.2.21:100;
    vrf-target target:1:2;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-MPLS-1.log size 10m;
                flag all;
            }
            extended-vlan-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
        }
```

```
    }
}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            routing-interface irb.1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            routing-interface irb.2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            routing-interface irb.3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            routing-interface irb.4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
            routing-interface irb.5;
            vxlan {
                vni 5;
            }
```

```
        }
    }
}
VRF {
    instance-type vrf;
    interface irb.1;
    interface irb.2;
    interface irb.3;
    interface irb.4;
    interface irb.5;
    route-distinguisher 1:1;
    vrf-target target:10:10;
}
```

Verifying Data Center Gateway and WAN Edge 4 Router (MX22) Configuration

Purpose Verify that MX22 is properly configured.

Action Verify that the interfaces on the MX11 router (DC GW/WAN Edge1) are configured for the following:

Underlay connectivity to the MX12, ToR11, ToR12, and P devices, which is the EVPN-VXLAN part of DC1 network.

```
user@MX11> show configuration interfaces
```

```
traceoptions {
    file R2-DCD.log size 10m;
    flag all;
}
ge-5/1/0 {
    unit 0 {
        description "CONNECTED TO MX-21";
        family inet {
            address 192.168.7.1/24;
        }
    }
}
ge-5/1/1 {
    unit 0 {
        description "CONNECTED TO TOR-11";
        family inet {
            address 192.168.3.2/24;
        }
    }
}
ge-5/1/8 {
    unit 0 {
        description "CONNECTED TO TOR-12";
        family inet {
            address 192.168.6.2/24;
        }
    }
}
ge-5/1/9 {
    unit 0 {
        description "CONNECTED TO P";
        family inet {
            address 203.0.1.1/24;
        }
        family mpls;
    }
}
```

Integrated routing and bridging (IRB) interfaces that advertises the MAC and IP routes (MAC+IP type 2 routes) for hosts in the topology. The IRB configuration is the gateway for the VLANs on the hosts.

```
user@MX11> show configuration interfaces
```

```
irb {
    unit 1 {
        proxy-macip-advertisement;
```

```
        virtual-gateway-esi {
            00:11:aa:aa:aa:aa:aa:aa:aa:aa;
            all-active;
        }
        family inet {
            address 10.11.1.12/24 {
                virtual-gateway-address 10.11.1.10;
            }
        }
    }
    unit 2 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:bb:bb:bb:bb:bb:bb:bb:bb;
            all-active;
        }
        family inet {
            address 10.12.1.12/24 {
                virtual-gateway-address 10.12.1.10;
            }
        }
    }
    unit 3 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:cc:cc:cc:cc:cc:cc:cc:cc;
            all-active;
        }
        family inet {
            address 10.13.1.12/24 {
                virtual-gateway-address 10.13.1.10;
            }
        }
    }
    unit 4 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:dd:dd:dd:dd:dd:dd:dd:dd;
            all-active;
        }
        family inet {
            address 10.14.1.12/24 {
                virtual-gateway-address 10.14.1.10;
            }
        }
    }
    unit 5 {
        proxy-macip-advertisement;
        virtual-gateway-esi {
            00:11:ee:ee:ee:ee:ee:ee:ee:ee;
            all-active;
        }
        family inet {
            address 10.15.1.12/24 {
                virtual-gateway-address 10.15.1.10;
            }
        }
    }
}
```

An ESI value and active-active multihoming on the logical tunnel interface. Use the same ESI value on all other gateway/WAN edge routers in the DC1 network.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    esi {
        00:22:22:22:22:22:22:22:22:22;
        all-active;
    }
}
```

A pair of logical tunnel (lt-) interface on the MX11 gateway router to interconnect the EVPN-VXLAN instance of the data center network with the MPLS-based EVPN instance of the WAN. One logical tunnel (lt-) interface is configured as the access interface for EVPN-VXLAN and the other logical tunnel (lt-) interface is configured as the access interface for MPLS-based EVPN.

```
user@MX11> show configuration interfaces
```

```
lt-5/1/0 {
    unit 0 {
        peer-unit 1;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
    unit 1 {
        peer-unit 0;
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
```

Loopback interface address.

```
user@MX11> show configuration interfaces
```

```
lo0 {
    unit 84 {
        family inet {
            address 192.0.2.21/32;
        }
        family mpls;
    }
}
```

Verify that the routing options and load balancing are properly configured.

```
user@MX11> show configuration routing-options
```

```
router-id 192.0.2.21;
autonomous-system 300;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and RSVP, MPLS, BGP, and OSPF protocols are properly configured.

```
user@MX11> show configuration protocols
```

```
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path MX11-TO-MX12 {
        to 192.0.2.22;
    }
    label-switched-path MX11-TO-P {
        to 203.0.113.1;
    }
    label-switched-path MX11-TO-MX21 {
        to 198.51.100.21;
    }
    label-switched-path MX11-TO-MX22 {
        to 198.51.100.22;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    local-address 192.0.2.21;
    local-as 300;
    group INT {
        type internal;
        local-address 192.0.2.21;
        family evpn {
            signaling;
        }
        export TEST;
        neighbor 203.0.113.1;
    }
    group TOR-11 {
        type external;
        local-address 192.168.3.2;
        import TEST;
        export [ TEST LO ];
        peer-as 100;
        local-as 400;
        neighbor 192.168.3.1 {
```

```
        family inet {
            unicast;
        }
    }
}
group TOR-12 {
    type external;
    local-address 192.168.6.2;
    export [ TEST LO ];
    peer-as 200;
    local-as 400;
    neighbor 192.168.6.1 {
        family inet {
            unicast;
        }
    }
}
group MX-12 {
    type external;
    local-address 192.168.7.1;
    export [ TEST LO ];
    peer-as 500;
    local-as 400;
    neighbor 192.168.7.2 {
        family inet {
            unicast;
        }
    }
}
group TOR-11-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 100;
    local-as 400;
    neighbor 192.0.2.11 {
        family evpn {
            signaling;
        }
    }
}
group TOR-12-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 192.0.2.21;
    export TEST;
    peer-as 200;
    local-as 400;
    neighbor 192.0.2.12 {
        family evpn {
            signaling;
        }
```

```
        }
    }
    group MX-12-EVPN {
        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 192.0.2.21;
        export TEST;
        peer-as 500;
        local-as 400;
        neighbor 192.0.2.22 {
            family evpn {
                signaling;
            }
        }
    }
    group MX-11-EVPN {
        export TEST;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-5/1/9.0;
        interface lo0.84 {
            passive;
        }
    }
}
l2-learning {
    traceoptions {
        file MX11-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@MX11> show configuration policy-options
```

```
policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 192.0.2.21/32 exact;
        }
        then accept;
    }
    from {
        protocol direct;
        route-filter 192.0.2.21/32 exact;
    }
```

```
    then accept;
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-based MPLS routing instances and EVPN-VXLAN routing instances are properly configured.

```
user@MX11> show configuration routing-instances
```

```

EVPN-MPLS-1 {
    instance-type virtual-switch;
    interface lt-5/1/0.0;
    route-distinguisher 192.0.2.21:100;
    vrf-target target:1:2;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-MPLS-1.log size 10m;
                flag all;
            }
            extended-vlan-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
        }
```

```
    }
}
EVPN-VXLAN-1 {
    vtep-source-interface lo0.84;
    instance-type virtual-switch;
    interface lt-5/1/0.1;
    route-distinguisher 192.0.2.21:1;
    vrf-target target:1:1;
    protocols {
        evpn {
            traceoptions {
                file MX11-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
            default-gateway no-gateway-community;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            routing-interface irb.1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            routing-interface irb.2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            routing-interface irb.3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            routing-interface irb.4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
            routing-interface irb.5;
            vxlan {
                vni 5;
            }
```

```
        }
    }
}
VRF {
    instance-type vrf;
    interface irb.1;
    interface irb.2;
    interface irb.3;
    interface irb.4;
    interface irb.5;
    route-distinguisher 1:1;
    vrf-target target:10:10;
}
```

Verifying ToR21 Configuration

Purpose Verify that ToR21 is properly configured.

Action Verify that the logical system interfaces and bridge domains on the CE4 device are properly configured to enable Layer 2 connectivity and to handle inter-VXLAN traffic.

```
user@ce4> show configuration logical-systems
```

```
CE-4 {
    interfaces {
        ge-1/0/9 {
            unit 0 {
                description "CONNECTED TO Host 4";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
        ge-1/1/6 {
            unit 0 {
                description "CONNECTED TO ToR21";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }

    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;

        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;

        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;

        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;

        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;

        }
    }
}
```

Verify that the interfaces and trace options on ToR21 are configured properly to enable

underlay connectivity to other ToR and gateway and WAN edge devices.

```
user@ToR21>show configuration interfaces
```

```
traceoptions {
    file R6-DCD.log size 10m;
    flag all;
}
xe-0/0/0 {
    unit 0 {
        description "CONNECTED TO MX22";
        family inet {
            address 192.168.10.2/24;
        }
    }
}
ge-1/0/0 {
    description "CONNECTED TO CE-5";
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/0/1 {
    unit 0 {
        description "CONNECTED TO MX21";
        family inet {
            address 192.168.101.1/24;
        }
    }
}
ge-1/0/6 {
    unit 0 {
        description "CONNECTED TO CE-4";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR22";
        family inet {
            address 192.168.12.1/24;
        }
    }
}
ae0 {
    esi {
        00:44:44:44:44:44:44:44:44:44;
        all-active;
    }
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 22:22:22:22:22:22;
        }
    }
```

```
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
lo0 {
    unit 90 {
        family inet {
            address 198.51.100.11/32;
        }
    }
}
```

Verify that the routing and load balancing options are properly configured.

```
user@ToR21> show configuration routing-options
```

```
router-id 198.51.100.11;
autonomous-system 600;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```
user@ToR21> show configuration protocols
```

```
bgp {
    export TEST;
    local-as 600;
    group MX21 {
        type external;
        local-address 192.168.9.2;
        export [ LO TEST ];
        peer-as 800;
        local-as 600;
        neighbor 192.168.9.1 {
            family inet {
                unicast;
            }
        }
    }
    group MX22 {
        type external;
        local-address 10.102.2.1;
        export [ LO TEST ];
        peer-as 900;
        local-as 600;
        neighbor 192.168.10.1 {
            family inet {
```

```
            unicast;
        }
    }
}
group ToR22 {
    type external;
    local-address 10.105.5.1;
    export [ LO TEST ];
    peer-as 700;
    local-as 600;
    neighbor 192.168.12.2 {
        family inet {
            unicast;
        }
    }
}
group MX21-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.11;
    peer-as 800;
    local-as 600;
    neighbor 198.51.100.21 {
        family evpn {
            signaling;
        }
    }
}
group MX22-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.11;
    peer-as 900;
    local-as 600;
    neighbor 198.51.100.22 {
        family evpn {
            signaling;
        }
    }
}
group ToR22-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.11;
    peer-as 700;
    local-as 600;
    neighbor 198.51.100.12 {
        family evpn {
            signaling;
        }
```

```
        }
    }
}
l2-learning {
    traceoptions {
        file TOR21-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@ToR21> show configuration policy-options
```

```
policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 198.51.100.11/32 exact;
        }
        then accept;
    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly configured.

```
user@ToR21> show configuration routing-instances
```

```
EVPN-VXLAN-1 {
    vtep-source-interface lo0.90;
    instance-type virtual-switch;
    interface ge-1/0/6.0;
    interface ae0.0;
    route-distinguisher 198.51.100.11:1;
    vrf-target target:1:3;
    protocols {
        evpn {
            traceoptions {
```

```
                file TOR21-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
            vxlan {
                vni 5;
            }
        }
    }
}
```

Verifying ToR22 Configuration

Purpose Verify that ToR22 is properly configured.

Action Verify that the logical system interfaces and bridge domains on the CE5 and CE6 devices are properly configured to enable Layer 2 connectivity and to handle inter-VXLAN traffic.

```
user@ce5> show configuration logical-systems
```

```
CE-5 {
    interfaces {
        ge-1/0/9 {
            unit 0 {
                description "CONNECTED TO Host 5";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
        ae1 {
            unit 0 {
                description "CONNECTED TO ToR21";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }

    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;

        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;

        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;

        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;

        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;

        }
    }
}
```

```
user@ce6> show configuration logical-systems
```

```
CE-6 {
    interfaces {
        ge-1/1/6 {
            unit 0 {
                description "CONNECTED TO ToR22";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
        }
        ge-1/1/9 {
            unit 0 {
                description "CONNECTED TO Host 6";
                family bridge {
                    interface-mode trunk;
                    vlan-id-list 1-5;
                }
            }
      }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;

        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;

        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;

        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;

        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;

        }
    }
}
```

Verify that the interfaces and trace options on ToR22 are configured properly to enable underlay connectivity to other ToR and gateway and WAN edge devices.

```
user@ToR22>show configuration interfaces
```

```
traceoptions {
```

```
    file R7-DCD.log size 10m;
    flag all;
}
xe-0/0/0 {
    unit 0 {
        description "CONNECTED TO MX22";
        family inet {
            address 192.168.11.2/24;
        }
    }
}
ge-1/0/0 {
    description "CONNECTED TO ToR21";
    gigether-options {
        802.3ad ae1;
    }
}
ge-1/0/6 {
    unit 0 {
        description "CONNECTED TO CE-6";
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ge-1/0/7 {
    description "CONNECTED TO ToR22";
    gigether-options {
        802.3ad ae1;
    }
}
ge-1/1/0 {
    unit 0 {
        description "CONNECTED TO MX21";
        family inet {
            address 192.168.8.2/24;
        }
    }
}
ge-1/1/3 {
    unit 0 {
        description "CONNECTED TO ToR21";
        family inet {
            address 192.168.12.2/24;
        }
    }
}
ge-1/1/7 {
    description "CONNECTED TO CE-5";
    gigether-options {
        802.3ad ae0;
    }
}
ae0 {
    esi {
        00:44:44:44:44:44:44:44:44:44;
        all-active;
    }
```

```
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 22:22:22:22:22:22;
        }
    }
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-5;
        }
    }
}
ae1 {
    aggregated-ether-options {
        lacp {
            active;
            periodic fast;
            system-id 22:22:22:22:22:22;
        }
    }
}
lo0 {
    unit 92 {
        family inet {
            address 198.51.100.12/32;
        }
    }
}
```

Verify that the routing and load balancing options are properly configured.

```
user@ToR22> show configuration routing-options
```

```


router-id 198.51.100.12;
autonomous-system 700;
forwarding-table {
    export evpn-pplb;
}
```

Verify that the multiprotocol external BGP (MP-EBGP) underlay and overlay protocols and Layer 2 address learning and forwarding properties are properly configured.

```
user@ToR22> show configuration protocols
```

```


bgp {
    export TEST;
    local-as 700;
    group MX21 {
        type external;
        local-address 192.168.8.2;
        export [ LO TEST ];
```

```
    peer-as 800;
    local-as 700;
    neighbor 192.168.8.1 {
        family inet {
            unicast;
        }
    }
}
group MX22 {
    type external;
    local-address 192.168.11.2;
    export [ LO TEST ];
    peer-as 900;
    local-as 700;
    neighbor 192.168.11.1 {
        family inet {
            unicast;
        }
    }
}
group ToR21 {
    type external;
    local-address 192.168.12.2;
    export [ LO TEST ];
    peer-as 600;
    local-as 700;
    neighbor 10.105.5.1 {
        family inet {
            unicast;
        }
    }
}
group MX21-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.12;
    peer-as 800;
    local-as 700;
    neighbor 198.51.100.21 {
        family evpn {
            signaling;
        }
    }
}
group MX22-EVPN {
    type external;
    multihop {
        ttl 2;
        no-nexthop-change;
    }
    local-address 198.51.100.12;
    peer-as 900;
    local-as 700;
    neighbor 198.51.100.22 {
        family evpn {
            signaling;
```

```
            }
        }
    }
    group ToR21-EVPN {
        type external;
        multihop {
            ttl 2;
            no-nexthop-change;
        }
        local-address 198.51.100.12;
        peer-as 600;
        local-as 700;
        neighbor 198.51.100.11 {
            family evpn {
                signaling;
            }
        }
    }
}
l2-learning {
    traceoptions {
        file TOR22-L2ALD.log size 10m;
        level all;
        flag all;
    }
}
```

Verify that the routing policies and community policy options and load balancing are properly configured to accept the direct loopback address route and redirect it into BGP.

```
user@ToR22> show configuration policy-options
```

```
policy-statement LO {
    term 1 {
        from {
            protocol direct;
            route-filter 198.51.100.12/32 exact;
        }
        then accept;
    }
}
policy-statement TEST {
    then {
        community add NO-EXPORT;
    }
}
policy-statement evpn-pplb {
    from protocol evpn;
    then {
        load-balance per-packet;
    }
}
community NO-EXPORT members [ no-advertise no-export no-export-subconfed ];
```

Verify that the EVPN-VXLAN routing instances for each virtual network are properly

configured.

```
user@ToR22> show configuration routing-instances
```

```
EVPN-VXLAN-1 {
    vtep-source-interface lo0.92;
    instance-type virtual-switch;
    interface ge-1/0/6.0;
    interface ae0.0;
    route-distinguisher 198.51.100.12:1;
    vrf-target target:1:3;
    protocols {
        evpn {
            traceoptions {
                file TOR22-EVPN-VXLAN-1.log size 10m;
                flag all;
            }
            encapsulation vxlan;
            extended-vni-list 1-5;
        }
    }
    bridge-domains {
        BD-1 {
            domain-type bridge;
            vlan-id 1;
            vxlan {
                vni 1;
            }
        }
        BD-2 {
            domain-type bridge;
            vlan-id 2;
            vxlan {
                vni 2;
            }
        }
        BD-3 {
            domain-type bridge;
            vlan-id 3;
            vxlan {
                vni 3;
            }
        }
        BD-4 {
            domain-type bridge;
            vlan-id 4;
            vxlan {
                vni 4;
            }
        }
        BD-5 {
            domain-type bridge;
            vlan-id 5;
            vxlan {
                vni 5;
            }
        }
    }
}
```

Related Documentation

- EVPN-VXLAN Data Center Interconnect Through EVPN-MPLS WAN Overview on page 743

CHAPTER 26

Extending a Junos Fusion Enterprise Using EVPN-MPLS

Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG

Starting with Junos OS Release 17.4R1, you can use Ethernet VPN (EVPN) to extend a Junos Fusion Enterprise or multichassis link aggregation group (MC-LAG) network over an MPLS network to a data center or campus network. With the introduction of this feature, you can now interconnect dispersed campus and data center sites to form a single Layer 2 virtual bridge.

Figure 87 on page 908 shows a Junos Fusion Enterprise topology with two EX9200 switches that serve as aggregation devices (PE2 and PE3) to which the satellite devices are multihomed. The two aggregation devices use an interchassis link (ICL) and the Inter-Chassis Control Protocol (ICCP) protocol from MC-LAG to connect and maintain the Junos Fusion Enterprise topology. PE1 in the EVPN-MPLS environment interworks with PE2 and PE3 in the Junos Fusion Enterprise with MC-LAG.

Figure 87: EVPN-MPLS Interworking with Junos Fusion Enterprise

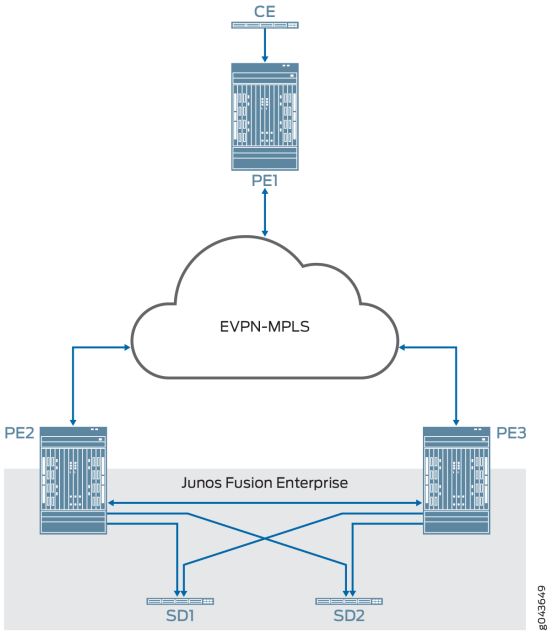
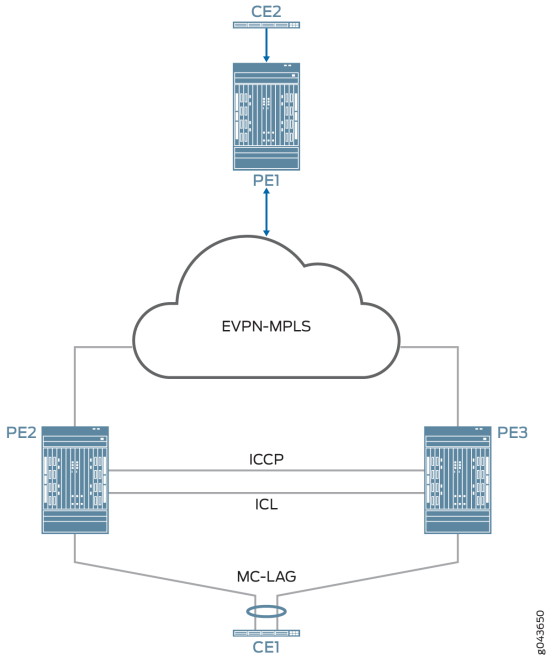


Figure 88 on page 909 shows an MC-LAG topology in which customer edge (CE) device CE1 is multihomed to PE2 and PE3. PE2 and PE3 use an ICL and the ICCP protocol from MC-LAG to connect and maintain the topology. PE1 in the EVPN-MPLS environment interworks with PE2 and PE3 in the MC-LAG environment.

Figure 88: EVPN-MPLS Interworking with MC-LAG



Throughout this topic, Figure 87 on page 908 and Figure 88 on page 909 serve as references to illustrate various scenarios and points.

The use cases depicted in Figure 87 on page 908 and Figure 88 on page 909 require the configuration of both EVPN multihoming in active-active mode and MC-LAG on PE2 and PE3. EVPN with multihoming active-active and MC-LAG have their own forwarding logic for handling traffic, in particular, broadcast, unknown unicast, and multicast (BUM) traffic. At times, the forwarding logic for EVPN with multihoming active-active and MC-LAG contradict each other and causes issues. This topic describes the issues and how the EVPN-MPLS interworking feature resolves these issues.

NOTE:

Other than the EVPN-MPLS interworking-specific implementations described in this topic, EVPN-MPLS, Junos Fusion Enterprise, and MC-LAG offer the same functionality and function the same as the standalone features.

- Benefits of Using EVPN-MPLS with Junos Fusion Enterprise and MC-LAG on page 910
- BUM Traffic Handling on page 910
- Split Horizon on page 910
- MAC Learning on page 911

Benefits of Using EVPN-MPLS with Junos Fusion Enterprise and MC-LAG

Use EVPN-MPLS with Junos Fusion Enterprise and MC-LAG to interconnect dispersed campus and data center sites to form a single Layer 2 virtual bridge.

BUM Traffic Handling

In the use cases shown in Figure 87 on page 908 and Figure 88 on page 909, PE1, PE2, and PE3 are EVPN peers, and PE2 and PE3 are MC-LAG peers. Both sets of peers exchange control information and forward traffic to each other, which causes issues. Table 19 on page 910 outlines the issues that arise, and how Juniper Networks resolves the issues in its implementation of the EVPN-MPLS interworking feature.

Table 19: BUM Traffic: Issues and Resolutions

BUM Traffic Direction	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach
North bound (PE2 receives BUM packet from a locally attached single- or dual-homed interfaces).	PE2 floods BUM packet to the following: • All locally attached interfaces, including the ICL, for a particular broadcast domain. • All remote EVPN peers for which PE2 has received inclusive multicast routes.	Between PE2 and PE3, there are two BUM forwarding paths—the MC-LAG ICL and an EVPN-MPLS path. The multiple forwarding paths result in packet duplication and loops.	• BUM traffic is forwarded on the ICL only. • Incoming traffic from the EVPN core is not forwarded on the ICL. • Incoming traffic from the ICL is not forwarded to the EVPN core.
South bound (PE1 forwards BUM packet to PE2 and PE3).	PE2 and PE3 both receive a copy of the BUM packet and flood the packet out of all of their local interfaces, including the ICL.	PE2 and PE3 both forward the BUM packet out of the ICL, which results in packet duplication and loops.	

Split Horizon

In the use cases shown in Figure 87 on page 908 and Figure 88 on page 909, split horizon prevents multiple copies of a BUM packet from being forwarded to a CE device (satellite device). However, the EVPN-MPLS and MC-LAG split horizon implementations contradict each other, which causes an issue. Table 20 on page 911 explains the issue and how Juniper Networks resolves it in its implementation of the EVPN-MPLS interworking feature.

Table 20: BUM Traffic: Split Horizon-Related Issue and Resolution

BUM Traffic Direction	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach

North bound (PE2 receives BUM packet from a locally attached dual-homed interface).	• Per EVPN-MPLS forwarding logic:
	  ◦ Only the designated forwarder (DF) for the Ethernet segment (ES) can forward BUM traffic.
	  ◦ The local bias rule, in which the local peer forwards the BUM packet and the remote peer drops it, is not supported.
	• Per MC-LAG forwarding logic, local bias is supported.
		The EVPN-MPLS and MC-LAG forwarding logic contradicts each other and can prevent BUM traffic from being forwarded to the ES.	Support local bias, thereby ignoring the DF and non-DF status of the port for locally switched traffic.

South bound (PE1 forwards BUM packet to PE2 and PE3).	Traffic received from PE1 follows the EVPN DF and non-DF forwarding rules for a mulithomed ES.	None.	Not applicable.

MAC Learning

EVPN and MC-LAG use the same method for learning MAC addresses—namely, a PE device learns MAC addresses from its local interfaces and synchronizes the addresses to its peers. However, given that both EVPN and MC-LAG are synchronizing the addresses, an issue arises.

Table 21 on page 911 describes the issue and how the EVPN-MPLS interworking implementation prevents the issue. The use cases shown in Figure 87 on page 908 and Figure 88 on page 909 illustrate the issue. In both use cases, PE1, PE2, and PE3 are EVPN peers, and PE2 and PE3 are MC-LAG peers.

Table 21: MAC Learning: EVPN and MC-LAG Synchronization Issue and Implementation Details

MAC Synchronization Use Case	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach
MAC addresses learned locally on single- or dual-homed interfaces on PE2 and PE3.	• Between the EVPN peers, MAC addresses are synchronized using the EVPN BGP control plane. • Between the MC-LAG peers, MAC addresses are synchronized using the MC-LAG ICCP control plane.	PE2 and PE3 function as both EVPN peers and MC-LAG peers, which result in these devices having multiple MAC synchronization paths.	• For PE1: use MAC addresses synchronized by EVPN BGP control plane. • For PE2 and PE3: use MAC addresses synchronized by MC-LAG ICCP control plane.

Table 21: MAC Learning: EVPN and MC-LAG Synchronization Issue and Implementation Details (continued)

MAC Synchronization Use Case	EVPN Interworking with Junos Fusion Enterprise and MC-LAG Logic	Issue	Juniper Networks Implementation Approach
MAC addresses learned locally on single- or dual-homed interfaces on PE1.	Between the EVPN peers, MAC addresses are synchronized using the EVPN BGP control plane.	None.	Not applicable.

Handling Down Link Between Cascade and Uplink Ports in Junos Fusion Enterprise

NOTE: **This section applies only to EVPN-MPLS interworking with a Junos Fusion Enterprise.**

In the Junos Fusion Enterprise shown in Figure 87 on page 908, assume that aggregation device PE2 receives a BUM packet from PE1 and that the link between the cascade port on PE2 and the corresponding uplink port on satellite device SD1 is down. Regardless of whether the BUM packet is handled by MC-LAG or EVPN multihoming active-active, the result is the same—the packet is forwarded via the ICL interface to PE3, which forwards it to dual-homed SD1.

To further illustrate how EVPN with multihoming active-active handles this situation with dual-homed SD1, assume that the DF interface resides on PE2 and is associated with the down link and that the non-DF interface resides on PE3. Typically, per EVPN with multihoming active-active forwarding logic, the non-DF interface drops the packet. However, because of the down link associated with the DF interface, PE2 forwards the BUM packet via the ICL to PE3, and the non-DF interface on PE3 forwards the packet to SD1.

Layer 3 Gateway Support

The EVPN-MPLS interworking feature supports the following Layer 3 gateway functionality for extended bridge domains and VLANs:

- Integrated routing and bridging (IRB) interfaces to forward traffic between the extended bridge domains or VLANs.
- Default Layer 3 gateways to forward traffic from a physical (bare-metal) server in an extended bridge domain or VLAN to a physical server or virtual machine in another extended bridge domain or VLAN.

Release History Table

Release	Description
17.4R1	Starting with Junos OS Release 17.4R1, you can use Ethernet VPN (EVPN) to extend a Junos Fusion Enterprise or multichassis link aggregation group (MC-LAG) network over an MPLS network to a data center or campus network.

Example: EVPN-MPLS Interworking With Junos Fusion Enterprise

This example shows how to use Ethernet VPN (EVPN) to extend a Junos Fusion Enterprise over an MPLS network to a geographically distributed campus or enterprise network.

EVPN-MPLS interworking is supported with a Junos Fusion Enterprise, which is based on a multichassis link aggregation group (MC-LAG) infrastructure to provide redundancy for the EX9200 switches that function as aggregation devices.

The aggregation devices in the Junos Fusion Enterprise are connected to a provider edge (PE) device in an MPLS network. The PE device can be either an MX Series router or an EX9200 switch.

This example shows how to configure the aggregation devices in the Junos Fusion Enterprise and the PE device in the MPLS network to interwork with each other.

Requirements

This example uses the following hardware and software components:

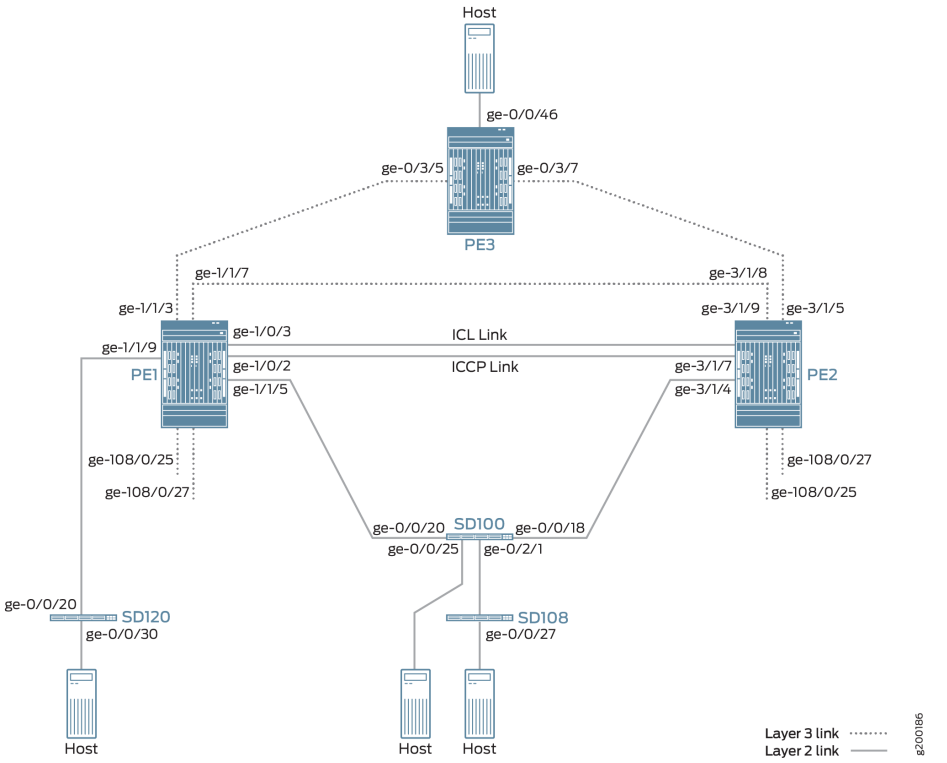
- Three EX9200 switches:
  - PE1 and PE2, which both function as aggregation devices in the Junos Fusion Enterprise and EVPN BGP peers in the EVPN-MPLS overlay network.
  - PE3, which functions as an EVPN BGP peer in the EVPN-MPLS overlay network.
- The EX9200 switches are running Junos OS Release 17.4R1 or later software.

NOTE: Although the Junos Fusion Enterprise includes three satellite devices, this example focuses on the configuration of the PE1, PE2, and PE3. For more information about configuring satellite devices, see *Configuring or Expanding a Junos Fusion Enterprise*.

Overview and Topology

Figure 89 on page 914 shows a Junos Fusion Enterprise with dual aggregation devices PE1 and PE2. The aggregation devices are connected using an interchassis link (ICL) and communicate with each other using the Inter-Chassis Control Protocol (ICCP).

Figure 89: EVPN-MPLS Interworking with Junos Fusion Enterprise



The Junos Fusion Enterprise also includes three satellite devices. Satellite device SD120 is a standalone satellite device that has a single-homed connection to PE1. Satellite devices SD100 and SD108 are included in a cluster named Cluster_100_108. SD100 is the only cluster member with a connection to an aggregation device, in this case, multhomed connections to PE1 and PE2.

The topology in Figure 89 on page 914 also includes PE3, which is positioned at the edge of an MPLS network. PE3 functions as the gateway between the Junos Fusion Enterprise network and a geographically distributed campus or enterprise network. PE1, PE2, and PE3 run EVPN, which enables hosts in the Junos Fusion Enterprise network to communicate with hosts in the campus or enterprise network by way of the intervening MPLS network.

From the perspective of the EVPN-MPLS interworking feature, PE3 functions solely as an EVPN BGP peer, and PE1 and PE2 in the Junos Fusion Enterprise have dual roles:

- Aggregation devices in the Junos Fusion Enterprise.

- EVPN BGP peers in the EVPN-MPLS network.

Because of the dual roles, PE1 and PE2 are configured with Junos Fusion Enterprise, EVPN, BGP, and MPLS attributes.

Table 22 on page 915 outlines key Junos Fusion Enterprise and EVPN (BGP and MPLS) attributes configured on PE1, PE2, and PE3.

Table 22: Key Junos Fusion Enterprise and EVPN (BGP and MPLS) Attributes Configured on PE1, PE2, and PE3

Key Attributes	PE1	PE2	PE3
Junos Fusion Enterprise Attributes			
Interfaces	ICL: ge-1/0/3 ICCP: ge-1/0/2	ICL: ge-3/1/9 ICCP: ge-3/1/7	Not applicable
EVPN-MPLS			
Interfaces	Connection to PE3: ge-1/1/3 Connection to PE2: ge-1/1/7	Connection to PE3: ge-3/1/5 Connection to PE1: ge-3/1/8	Connection to PE1: ge-0/3/5 Connection to PE2: ge-0/3/7
IP addresses	BGP peer address: 10.25.0.1	BGP peer address: 10.25.0.2	BGP peer address: 10.25.0.3
Autonomous system	100	100	100
Virtual switch routing instances	evpn1	evpn1	evpn1

Note the following about the EVPN-MPLS interworking feature and its configuration:

- You must configure Ethernet segment identifiers (ESIs) on the dual-homed extended ports in the Junos Fusion Enterprise. The ESIs enable EVPN to identify the dual-homed extended ports.
- The only type of routing instance that is supported is the virtual switch instance (**set routing-instances *name* instance-type virtual-switch**).
- Only one virtual switch instance is supported with Junos Fusion Enterprise.
- On the aggregation devices in the Junos Fusion Enterprise, you must include the bgp-peer configuration statement in the **[edit routing-instances *name* protocols evpn mclag]** hierarchy level. This configuration statement enables the interworking of EVPN-MPLS with Junos Fusion Enterprise on the aggregation devices.
- Address Resolution Protocol (ARP) suppression is not supported.

Aggregation Device (PE1 and PE2) Configuration

To configure aggregation devices PE1 and PE2, perform these tasks.

NOTE: This section focuses on enabling EVPN-MPLS on PE1 and PE2. As a result, the Junos Fusion Enterprise configuration on PE1 and PE2 is performed without the use of the configuration synchronization feature. For information about configuration synchronization, see *Understanding Configuration Synchronization*.

CLI Quick Configuration

PE1: Junos Fusion Enterprise Configuration

```
set interfaces ge-1/1/9 cascade-port
set interfaces ge-1/1/5 cascade-port
set chassis satellite-management fpc 120 cascade-ports ge-1/1/9
set chassis satellite-management cluster Cluster_100_108 cluster-id 2
set chassis satellite-management cluster Cluster_100_108 cascade-ports ge-1/1/5
set chassis satellite-management cluster Cluster_100_108 fpc 100 alias SD100
set chassis satellite-management cluster Cluster_100_108 fpc 100 system-id
  88:e0:f3:1f:3d:50
set chassis satellite-management cluster Cluster_100_108 fpc 108 alias SD108
set chassis satellite-management cluster Cluster_100_108 fpc 108 system-id
  88:e0:f3:1f:c8:d1
set chassis satellite-management cluster Cluster_100_108 fpc 100 member-id 1
set chassis satellite-management cluster Cluster_100_108 fpc 108 member-id 8
set chassis satellite-management upgrade-groups upgrade_120 satellite 120
set chassis satellite-management upgrade-groups upgrade_100 satellite 100
set chassis satellite-management redundancy-groups rg1 redundancy-group-id 2
set chassis satellite-management redundancy-groups chassis-id 1
set chassis satellite-management redundancy-groups rg1 peer-chassis-id 2
  inter-chassis-link ge-1/0/3
set chassis satellite-management redundancy-groups rg1 cluster Cluster_100_108
set interfaces ge-1/0/2 description iccp-link
set interfaces ge-1/0/2 unit 0 family inet address 10.20.20.1/24
set interfaces ge-1/0/3 description icl-link
set interfaces ge-1/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-1/0/3 unit 0 family ethernet-switching vlan members 100
set switch-options service-id 1
```

PE1: EVPN-MPLS Configuration

```
set interfaces lo0 unit 0 family inet address 10.25.0.1/32
set interfaces ge-1/1/3 unit 0 family inet address 10.0.1.1/30
set interfaces ge-1/1/3 unit 0 family mpls
set interfaces ge-1/1/7 unit 0 family inet address 10.0.3.1/30
set interfaces ge-1/1/7 unit 0 family mpls
set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
set interfaces ge-108/0/25 unit 0 esi all-active
set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan members v100
```

```
set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
set interfaces ge-108/0/27 unit 0 esi all-active
set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan members v100
set routing-options router-id 10.25.0.1
set routing-options autonomous-system 100
set protocols mpls interface lo0.0
set protocols mpls interface ge-1/1/3.0
set protocols mpls interface ge-1/1/7.0
set protocols bgp local-address 10.25.0.1
set protocols bgp peer-as 100
set protocols bgp local-as 100
set protocols bgp group evpn-mes type internal
set protocols bgp group evpn-mes family evpn signaling
set protocols bgp group evpn-mes peer-as 100
set protocols bgp group evpn-mes neighbor 10.25.0.2
set protocols bgp group evpn-mes neighbor 10.25.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/1/3.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-1/1/7.0
set protocols ldp interface lo0.0
set protocols ldp interface ge-1/1/3.0
set protocols ldp interface ge-1/1/7.0
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface ge-108/0/25.0
set routing-instances evpn1 interface ge-108/0/27.0
set routing-instances evpn1 interface ge-1/0/3.0
set routing-instances evpn1 route-distinguisher 10.25.0.1:1
set routing-instances evpn1 vrf-target target:100:1
set routing-instances evpn1 protocols evpn label-allocation per-instance
set routing-instances evpn1 protocols evpn extended-vlan-list 100
set routing-instances evpn1 protocols evpn mclag bgp-peer 10.25.0.2
set routing-instances evpn1 switch-options service-id 2
set routing-instances evpn1 vlans v100 vlan-id 100
```

PE2: Junos Fusion Enterprise Configuration

```
set interfaces ge-3/1/4 cascade-port
set chassis satellite-management cluster Cluster_100_108 cluster-id 2
set chassis satellite-management cluster Cluster_100_108 cascade-ports ge-3/1/4
set chassis satellite-management cluster Cluster_100_108 fpc 100 alias SD100
set chassis satellite-management cluster Cluster_100_108 fpc 100 system-id
  88:e0:f3:1f:3d:50
set chassis satellite-management cluster Cluster_100_108 fpc 108 alias SD108
set chassis satellite-management cluster Cluster_100_108 fpc 108 system-id
  88:e0:f3:1f:c8:d1
set chassis satellite-management cluster Cluster_100_108 fpc 100 member-id 1
set chassis satellite-management cluster Cluster_100_108 fpc 108 member-id 8
set chassis satellite-management upgrade-groups upgrade_100 satellite 100
set chassis satellite-management redundancy-groups rg1 redundancy-group-id 2
set chassis satellite-management redundancy-groups chassis-id 2
set chassis satellite-management redundancy-groups rg1 peer-chassis-id 1
  inter-chassis-link ge-3/1/9
```

```
set chassis satellite-management redundancy-groups rg1 cluster Cluster_100_108
set interfaces ge-3/1/7 description iccp-link
set interfaces ge-3/1/7 unit 0 family inet address 10.20.20.2/24
set interfaces ge-3/1/9 description icl-link
set interfaces ge-3/1/9 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-3/1/9 unit 0 family ethernet-switching vlan members 100
set switch-options service-id 1
```

PE2: EVPN-MPLS Configuration

```
set interfaces lo0 unit 0 family inet address 10.25.0.2/32
set interfaces ge-3/1/5 unit 0 family inet address 10.0.4.2/30
set interfaces ge-3/1/5 unit 0 family mpls
set interfaces ge-3/1/8 unit 0 family inet address 10.0.3.2/30
set interfaces ge-3/1/8 unit 0 family mpls
set interfaces irb unit 0 family inet address 10.5.5.1/24 virtual-gateway-address 10.5.5.5
set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
set interfaces ge-108/0/25 unit 0 esi all-active
set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan members v100
set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
set interfaces ge-108/0/27 unit 0 esi all-active
set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan members v100
set routing-options router-id 10.25.0.2
set routing-options autonomous-system 100
set protocols mpls interface lo0.0
set protocols mpls interface ge-3/1/5.0
set protocols mpls interface ge-3/1/8.0
set protocols bgp local-address 10.25.0.2
set protocols bgp peer-as 100
set protocols bgp local-as 100
set protocols bgp group evpn-mes type internal
set protocols bgp group evpn-mes family evpn signaling
set protocols bgp group evpn-mes peer-as 100
set protocols bgp group evpn-mes neighbor 10.25.0.1
set protocols bgp group evpn-mes neighbor 10.25.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-3/1/5.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-3/1/8.0
set protocols ldp interface lo0.0
set protocols ldp interface ge-3/1/5.0
set protocols ldp interface ge-3/1/8.0
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface ge-108/0/25.0
set routing-instances evpn1 interface ge-108/0/27.0
set routing-instances evpn1 interface ge-3/1/9.0
set routing-instances evpn1 route-distinguisher 10.25.0.2:1
set routing-instances evpn1 vrf-target target:100:1
set routing-instances evpn1 protocols evpn label-allocation per-instance
set routing-instances evpn1 protocols evpn extended-vlan-list 100
set routing-instances evpn1 protocols evpn mclag bgp-peer 10.25.0.1
set routing-instances evpn1 switch-options service-id 2
set routing-instances evpn1 vlans v100 vlan-id 100
```

```
set routing-instances evpn1 vlans v100 l3-interface irb.0
set routing-instances evpn1 vlans v100 no-arp-suppression
```

PE1: Configuring Junos Fusion Enterprise

Step-by-Step Procedure

1. Configure the cascade ports.

```
[edit]
user@switch# set interfaces ge-1/1/9 cascade-port
user@switch# set interfaces ge-1/1/5 cascade-port
```

2. Configure the FPC slot ID for standalone satellite device SD120 and map it to a cascade port.

```
[edit]
user@switch# set chassis satellite-management fpc 120 cascade-ports ge-1/1/9
```

3. Create a satellite device cluster, and assign a name and a cluster ID to it.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 cluster-id
  2
```

4. Define the cascade ports associated with the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108
  cascade-ports ge-1/1/5
user@switch# set chassis satellite-management cluster Cluster_100_108
  cascade-ports ge-1/1/9
```

5. Configure the FPC slot ID number, and map it to the MAC address of satellite devices SD100 and SD108, respectively.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100
  alias SD100
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100
  system-id 88:e0:f3:1f:3d:50
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108
  alias SD108
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108
  system-id 88:e0:f3:1f:c8:d1
```

6. Assign a member ID to each satellite device in the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100
  member-id 1
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108
  member-id 8
```

7. Create two satellite software upgrade groups—one that includes satellite device SD120 and another that includes satellite device SD100.

```
[edit]
user@switch# set chassis satellite-management upgrade-groups upgrade_120
  satellite 120
user@switch# set chassis satellite-management upgrade-groups upgrade_100
  satellite 100
```

8. Create and configure a redundancy group, which includes the aggregation devices and satellite devices in Cluster_100_108.

```
[edit]
user@switch# set chassis satellite-management redundancy-groups rg1
  redundancy-group-id 2
user@switch# set chassis satellite-management redundancy-groups chassis-id 1
user@switch# set chassis satellite-management redundancy-groups rg1
  peer-chassis-id 2 inter-chassis-link ge-1/0/3
user@switch# set chassis satellite-management redundancy-groups rg1 cluster
  Cluster_100_108
```

9. Configure the ICL and ICCP links.

```
[edit]
user@switch# set interfaces ge-1/0/2 description iccp-link
user@switch# set interfaces ge-1/0/2 unit 0 family inet address 10.20.20.1/24
user@switch# set interfaces ge-1/0/3 description icl-link
user@switch# set interfaces ge-1/0/3 unit 0 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces ge-1/0/3 unit 0 family ethernet-switching vlan members
  100
user@switch# set switch-options service-id 1
```

NOTE: While this step shows the configuration of interface ge-1/0/2, which is designated as the ICCP interface, it does not show how to configure the ICCP attributes on interface ge-1/0/2. By default, ICCP is automatically provisioned in a Junos Fusion Enterprise using dual aggregation devices. For more information about the automatic provisioning of ICCP, see *Configuring or Expanding a Junos Fusion Enterprise*.

PE1: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface and the interfaces connected to the other PE devices.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 10.25.0.1/32
user@switch# set interfaces ge-1/1/3 unit 0 family inet address 10.0.1.1/30
user@switch# set interfaces ge-1/1/3 unit 0 family mpls
user@switch# set interfaces ge-1/1/7 unit 0 family inet address 10.0.3.1/30
user@switch# set interfaces ge-1/1/7 unit 0 family mpls
```

2. Configure the extended ports with EVPN multihoming in active-active mode, an ESI, and map the ports to VLAN v100..

```
[edit]
user@switch# set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
user@switch# set interfaces ge-108/0/25 unit 0 esi all-active
user@switch# set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan
  members v100
user@switch# set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
user@switch# set interfaces ge-108/0/27 unit 0 esi all-active
user@switch# set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan
  members v100
```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 10.25.0.1
user@switch# set routing-options autonomous-system 100
```

4. Enable MPLS on the loopback interface and interfaces ge-1/1/3.0 and ge-1/1/7.0.

```
[edit]
user@switch# set protocols mpls interface lo0.0
user@switch# set protocols mpls interface ge-1/1/3.0
user@switch# set protocols mpls interface ge-1/1/7.0
```

5. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
user@switch# set protocols bgp local-address 10.25.0.1
user@switch# set protocols bgp peer-as 100
user@switch# set protocols bgp local-as 100
user@switch# set protocols bgp group evpn-mes type internal
user@switch# set protocols bgp group evpn-mes family evpn signaling
user@switch# set protocols bgp group evpn-mes peer-as 100
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.2
```

```
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.3
```

6. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf traffic-engineering
user@switch# set protocols ospf area 0.0.0.0 interface ge-1/1/3.0
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface fxp0.0 disable
user@switch# set protocols ospf area 0.0.0.0 interface ge-1/1/7.0
```

7. Configure the Label Distribution Protocol (LDP) on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface ge-1/1/3.0
user@switch# set protocols ldp interface ge-1/1/7.0
```

8. Configure a virtual switch routing instance for VLAN v100, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface ge-108/0/25.0
user@switch# set routing-instances evpn1 interface ge-108/0/27.0
user@switch# set routing-instances evpn1 interface ge-1/0/3.0
user@switch# set routing-instances evpn1 route-distinguisher 10.25.0.1:1
user@switch# set routing-instances evpn1 vrf-target target:100:1
user@switch# set routing-instances evpn1 protocols evpn label-allocation
   per-instance
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 100
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer 10.25.0.2
user@switch# set routing-instances evpn1 switch-options service-id 2
user@switch# set routing-instances evpn1 vlans v100 vlan-id 100
```

PE2: Configuring Junos Fusion Enterprise

Step-by-Step Procedure

1. Configure the cascade port.

```
[edit]
user@switch# set interfaces ge-3/1/4 cascade-port
```

2. Create a satellite device cluster, and assign a name and a cluster ID to it.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 cluster-id
  2
```

3. Define the cascade port associated with the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108
  cascade-ports ge-3/1/4
```

4. Configure the FPC slot ID number, and map it to the MAC address of satellite devices SD100 and SD108, respectively.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100
  alias SD100
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100
  system-id 88:e0:f3:1f:3d:50
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108
  alias SD108
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108
  system-id 88:e0:f3:1f:c8:d1
```

5. Assign a member ID to each satellite device in the satellite device cluster.

```
[edit]
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 100
  member-id 1
user@switch# set chassis satellite-management cluster Cluster_100_108 fpc 108
  member-id 8
```

6. Create a satellite software upgrade group that includes satellite device SD100.

```
[edit]
user@switch# set chassis satellite-management upgrade-groups upgrade_100
  satellite 100
```

7. Create and configure a redundancy group, which includes the aggregation devices and satellite devices in Cluster_100_108.

```
[edit]
user@switch# set chassis satellite-management redundancy-groups rg1
  redundancy-group-id 2
user@switch# set chassis satellite-management redundancy-groups chassis-id 2
```

```
user@switch# set chassis satellite-management redundancy-groups rg1
  peer-chassis-id 1inter-chassis-link ge-3/1/9
user@switch# set chassis satellite-management redundancy-groups rg1 cluster
  Cluster_100_108
```

8. Configure the ICL and ICCP links.

```
[edit]
user@switch# set interfaces ge-3/1/7 description iccp-link
user@switch# set interfaces ge-3/1/7 unit 0 family inet address 10.20.20.2/24
user@switch# set interfaces ge-3/1/9 description icl-link
user@switch# set interfaces ge-3/1/9 unit 0 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces ge-3/1/9 unit 0 family ethernet-switching vlan members
  100
user@switch# set switch-options service-id 1
```

NOTE: While this step shows the configuration of interface ge-3/1/7, which is designated as the ICCP interface, it does not show how to configure the ICCP attributes on interface ge-3/1/7. By default, ICCP is automatically provisioned in a Junos Fusion Enterprise using dual aggregation devices. For more information about the automatic provisioning of ICCP, see *Configuring or Expanding a Junos Fusion Enterprise*.

PE2: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, the interfaces connected to the other PE devices, and an IRB interface that is also configured as a default Layer 3 gateway.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 10.25.0.2/32
user@switch# set interfaces ge-3/1/5 unit 0 family inet address 10.0.4.2/30
user@switch# set interfaces ge-3/1/5 unit 0 family mpls
user@switch# set interfaces ge-3/1/8 unit 0 family inet address 10.0.3.2/30
user@switch# set interfaces ge-3/1/8 unit 0 family mpls
user@switch# set interfaces irb unit 0 family inet address 10.5.5.1/24
  virtual-gateway-address 10.5.5.5
```

2. Configure the extended ports with EVPN multihoming in active-active mode, an ESI, and map the ports to VLAN v100..

```
[edit]
user@switch# set interfaces ge-108/0/25 unit 0 esi 00:01:02:03:04:00:01:02:04:26
user@switch# set interfaces ge-108/0/25 unit 0 esi all-active
```

```
user@switch# set interfaces ge-108/0/25 unit 0 family ethernet-switching vlan
  members v100
user@switch# set interfaces ge-108/0/27 unit 0 esi 00:01:02:03:04:00:01:02:04:28
user@switch# set interfaces ge-108/0/27 unit 0 esi all-active
user@switch# set interfaces ge-108/0/27 unit 0 family ethernet-switching vlan
  members v100
```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 10.25.0.2
user@switch# set routing-options autonomous-system 100
```

4. Enable MPLS on the loopback interface and interfaces ge-3/1/5.0 and ge-3/1/8.0.

```
[edit]
user@switch# set protocols mpls interface lo0.0
user@switch# set protocols mpls interface ge-3/1/5.0
user@switch# set protocols mpls interface ge-3/1/8.0
```

5. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
user@switch# set protocols bgp local-address 10.25.0.2
user@switch# set protocols bgp peer-as 100
user@switch# set protocols bgp local-as 100
user@switch# set protocols bgp group evpn-mes type internal
user@switch# set protocols bgp group evpn-mes family evpn signaling
user@switch# set protocols bgp group evpn-mes peer-as 100
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.1
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.3
```

6. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf traffic-engineering
user@switch# set protocols ospf area 0.0.0.0 interface ge-3/1/5.0
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface fxp0.0 disable
user@switch# set protocols ospf area 0.0.0.0 interface ge-3/1/8.0
```

7. Configure the LDP on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
```

```
user@switch# set protocols ldp interface ge-3/1/5.0
user@switch# set protocols ldp interface ge-3/1/8.0
```

8. Configure a virtual switch routing instance for VLAN v100, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface ge-108/0/25.0
user@switch# set routing-instances evpn1 interface ge-108/0/27.0
user@switch# set routing-instances evpn1 interface ge-3/1/9.0
user@switch# set routing-instances evpn1 route-distinguisher 10.25.0.2:1
user@switch# set routing-instances evpn1 vrf-target target:100:1
user@switch# set routing-instances evpn1 protocols evpn label-allocation
  per-instance
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 100
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer 10.25.0.1
user@switch# set routing-instances evpn1 switch-options service-id 2
user@switch# set routing-instances evpn1 vlans v100 vlan-id 100
user@switch# set routing-instances evpn1 vlans v100 l3-interface irb.0
user@switch# set routing-instances evpn1 vlans v100 no-arp-suppression
```

PE3 Configuration

CLI Quick Configuration

PE3: EVPN-MPLS Configuration

```
set interfaces lo0 unit 0 family inet address 10.25.0.3/32
set interfaces ge-0/3/5 unit 0 family inet address 10.0.1.2/30
set interfaces ge-0/3/5 unit 0 family mpls
set interfaces ge-0/3/7 unit 0 family inet address 10.0.4.1/30
set interfaces ge-0/3/7 unit 0 family mpls
set interfaces ge-0/0/46 unit 0 esi 00:01:02:03:04:00:01:02:04:12
set interfaces ge-0/0/46 unit 0 esi all-active
set interfaces ge-0/0/46 unit 0 family ethernet-switching vlan members 100
set routing-options router-id 10.25.0.3
set routing-options autonomous-system 100
set routing-options forwarding-table export evpn-pplb
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/3/5.0
set protocols mpls interface ge-0/3/7.0
set protocols bgp local-address 10.25.0.3
set protocols bgp peer-as 100
set protocols bgp local-as 100
set protocols bgp group evpn-mes type internal
set protocols bgp group evpn-mes family evpn signaling
set protocols bgp group evpn-mes peer-as 100
set protocols bgp group evpn-mes neighbor 10.25.0.2
set protocols bgp group evpn-mes neighbor 10.25.0.1
set protocols ospf traffic-engineering
```

```
set protocols ospf area 0.0.0.0 interface ge-0/3/5.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-0/3/7.0
set protocols ldp interface lo0.0
set protocols ldp interface ge-0/3/5.0
set protocols ldp interface ge-0/3/7.0
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface ge-0/0/46.0
set routing-instances evpn1 route-distinguisher 10.25.0.3:1
set routing-instances evpn1 vrf-target target:100:1
set routing-instances evpn1 protocols evpn label-allocation per-instance
set routing-instances evpn1 protocols evpn extended-vlan-list 100
set routing-instances evpn1 switch-options service-id 2
set routing-instances evpn1 vlans v100 vlan-id 100
```

PE3: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the interfaces on EVPN-MPLS interworking occurs.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 10.25.0.3/32
user@switch# set interfaces ge-0/3/5 unit 0 family inet address 10.0.1.2/30
user@switch# set interfaces ge-0/3/5 unit 0 family mpls
user@switch# set interfaces ge-0/3/7 unit 0 family inet address 10.0.4.1/30
user@switch# set interfaces ge-0/3/7 unit 0 family mpls
```

2. Configure interface ge-0/0/46 with EVPN multihoming in active-active mode, an ESI, and map the ports to VLAN v100..

```
[edit]
user@switch# set interfaces ge-0/0/46 unit 0 esi 00:01:02:03:04:00:01:02:04:12
user@switch# set interfaces ge-0/0/46 unit 0 esi all-active
user@switch# set interfaces ge-0/0/46 unit 0 family ethernet-switching vlan
   members 100
```

3. Assign a router ID and the autonomous system in which the PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 10.25.0.2
user@switch# set routing-options autonomous-system 100
```

4. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```
[edit]
user@switch# set routing-options forwarding-table export evpn-pplb
user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
```

```
user@switch# set policy-options policy-statement evpn-pplb then load-balance
  per-packet
```

5. Enable MPLS on the loopback interface and interfaces ge-0/3/5.0 and ge-0/3/7.0.

```
[edit]
user@switch# set protocols mpls interface lo0.0
user@switch# set protocols mpls interface ge-0/3/5.0
user@switch# set protocols mpls interface ge-0/3/7.0
```

6. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
user@switch# set protocols bgp local-address 10.25.0.3
user@switch# set protocols bgp peer-as 100
user@switch# set protocols bgp local-as 100
user@switch# set protocols bgp group evpn-mes type internal
user@switch# set protocols bgp group evpn-mes family evpn signaling
user@switch# set protocols bgp group evpn-mes peer-as 100
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.2
user@switch# set protocols bgp group evpn-mes neighbor 10.25.0.1
```

7. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf traffic-engineering
user@switch# set protocols ospf area 0.0.0.0 interface ge-0/3/5.0
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface fxp0.0 disable
user@switch# set protocols ospf area 0.0.0.0 interface ge-0/3/7.0
```

8. Configure the LDP on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface ge-0/3/5.0
user@switch# set protocols ldp interface ge-0/3/7.0
```

9. Configure a virtual switch routing instance for VLAN v100, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface ge-0/0/46.0
user@switch# set routing-instances evpn1 route-distinguisher 10.25.0.3:1
```

```
user@switch# set routing-instances evpn1 vrf-target target:100:1
user@switch# set routing-instances evpn1 protocols evpn label-allocation
   per-instance
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 100
user@switch# set routing-instances evpn1 switch-options service-id 2
user@switch# set routing-instances evpn1 vlans v100 vlan-id 100
```

Related Documentation

- Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG on page 907

Example: EVPN-MPLS Interworking With an MC-LAG Topology

This example shows how to use Ethernet VPN (EVPN) to extend a multichassis link aggregation (MC-LAG) network over an MPLS network to a data center network or geographically distributed campus network.

EVPN-MPLS interworking is supported with an MC-LAG topology in which two MX Series routers, two EX9200 switches, or a mix of the two Juniper Networks devices function as MC-LAG peers, which use the Inter-Chassis Control Protocol (ICCP) and an interchassis link (ICL) to connect and maintain the topology. The MC-LAG peers are connected to a provider edge (PE) device in an MPLS network. The PE device can be either an MX Series router or an EX9200 switch.

This example shows how to configure the MC-LAG peers and PE device in the MPLS network to interwork with each other.

Requirements

This example uses the following hardware and software components:

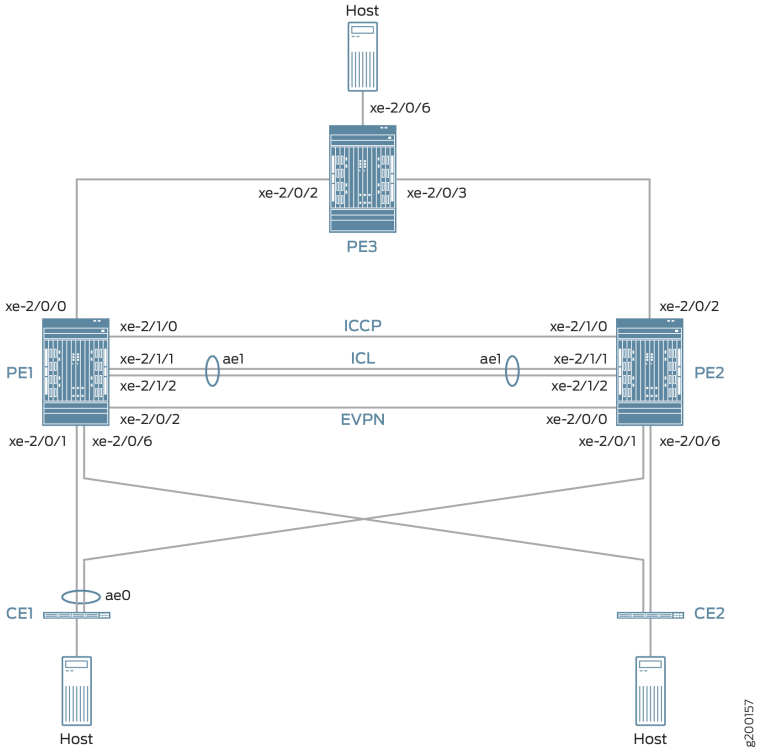
- Three EX9200 switches:
  - PE1 and PE2, which both function as MC-LAG peers in the MC-LAG topology and EVPN BGP peers in the EVPN-MPLS overlay network.
  - PE3, which functions as an EVPN BGP peer in the EVPN-MPLS overlay network.
- The EX9200 switches are running Junos OS Release 17.4R1 or later software.

NOTE: Although the MC-LAG topology includes two customer edge (CE) devices, this example focuses on the configuration of the PE1, PE2, and PE3.

Overview and Topology

Figure 90 on page 930 shows an MC-LAG topology with provider edge devices PE1 and PE2 that are configured as MC-LAG peers. The MC-LAG peers exchange control information over an ICCP link and data traffic over an ICL. In this example, the ICL is an aggregated Ethernet interface that is comprised of two interfaces.

Figure 90: EVPN-MPLS Interworking With an MC-LAG Topology



The topology in Figure 90 on page 930 also includes CE devices CE1 and CE2, which are both multihomed to each PE device. The links between CE1 and the two PE devices are bundled as an aggregated Ethernet interface on which MC-LAG in active-active mode is configured.

The topology in Figure 90 on page 930 also includes PE3 at the edge of an MPLS network. PE3 functions as the gateway between the MC-LAG network and either a data center or a geographically distributed campus network. PE1, PE2, and PE3 run EVPN, which enables hosts in the MC-LAG network to communicate with hosts in the data center or other campus network by way of an intervening MPLS network.

From the perspective of the EVPN-MPLS interworking feature, PE3 functions solely as an EVPN BGP peer, and PE1 and PE2 in the MC-LAG topology have dual roles:

- MC-LAG peers in the MC-LAG network.

- EVPN BGP peers in the EVPN-MPLS network.

Because of the dual roles, PE1 and PE2 are configured with MC-LAG, EVPN, BGP, and MPLS attributes.

Table 22 on page 915 outlines key MC-LAG and EVPN (BGP and MPLS) attributes configured on PE1, PE2, and PE3.

Table 23: Key MC-LAG and EVPN (BGP and MPLS) Attributes Configured on PE1, PE2, and PE3

Key Attributes	PE1	PE2	PE3
MC-LAG Attributes			
Interfaces	ICL: aggregated Ethernet interface ae1, which is comprised of xe-2/1/1 and xe-2/1/2 ICCP: xe-2/1/0	ICL: aggregated Ethernet interface ae1, which is comprised of xe-2/1/1 and xe-2/1/2 ICCP: xe-2/1/0	Not applicable
EVPN-MPLS			
Interfaces	Connection to PE3: xe-2/0/0 Connection to PE2: xe-2/0/2	Connection to PE3: xe-2/0/2 Connection to PE1: xe-2/0/0	Connection to PE1: xe-2/0/2 Connection to PE2: xe-2/0/3
IP addresses	BGP peer address: 198.51.100.1	BGP peer address: 198.51.100.2	BGP peer address: 198.51.100.3
Autonomous system	65000	65000	65000
Virtual switch routing instances	evpn1, evpn2, evpn3	evpn1, evpn2, evpn3	evpn1, evpn2, evpn3

Note the following about the EVPN-MPLS interworking feature and its configuration:

- You must configure Ethernet segment identifiers (ESIs) on the dual-homed interfaces in the MC-LAG topology. The ESIs enable EVPN to identify the dual-homed interfaces.
- The only type of routing instance that is supported is the virtual switch instance (**set routing-instances *name* instance-type virtual-switch**).
- On the MC-LAG peers, you must include the **bgp-peer** configuration statement in the **[edit routing-instances *name* protocols evpn mclag]** hierarchy level. This configuration statement enables the interworking of EVPN-MPLS with MC-LAG on the MC-LAG peers.
- Address Resolution Protocol (ARP) suppression is not supported.

PE1 and PE2 Configuration

To configure PE1 and PE2, perform these tasks:

- PE1: Configuring MC-LAG on page 936
- PE1: Configuring EVPN-MPLS on page 938
- PE2: Configuring MC-LAG on page 941
- PE2: Configuring EVPN-MPLS on page 943

CLI Quick Configuration

PE1: MC-LAG Configuration

```
set chassis aggregated-devices ethernet device-count 3
set interfaces xe-2/0/1 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
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 00:00:11:11:11:11
set interfaces ae0 aggregated-ether-options lacp admin-key 1
set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
set interfaces ae0 aggregated-ether-options mc-ae redundancy-group 2
set interfaces ae0 aggregated-ether-options mc-ae chassis-id 0
set interfaces ae0 aggregated-ether-options mc-ae mode active-active
set interfaces ae0 aggregated-ether-options mc-ae status-control active
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces ae0 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 1 family ethernet-switching vlan members 1
set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 unit 2 esi all-active
set interfaces ae0 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 2 family ethernet-switching vlan members 2
set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
set interfaces ae0 unit 3 esi all-active
set interfaces ae0 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 3 family ethernet-switching vlan members 3
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.1/24
set interfaces xe-2/1/1 gigether-options 802.3ad ae1
set interfaces xe-2/1/2 gigether-options 802.3ad ae1
set interfaces ae1 flexible-vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 unit 1 family ethernet-switching interface-mode trunk
```

```
set interfaces ae1 unit 1 family ethernet-switching vlan members 1
set interfaces ae1 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 2 family ethernet-switching vlan members 2
set interfaces ae1 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 3 family ethernet-switching vlan members 3
set multi-chassis multi-chassis-protection 203.0.113.2 interface ae1
set protocols iccp local-ip-addr 203.0.113.1
set protocols iccp peer 203.0.113.2 session-establishment-hold-time 600
set protocols iccp peer 203.0.113.2 redundancy-group-id-list 2
set protocols iccp peer 203.0.113.2 liveness-detection minimum-interval 10000
set protocols iccp peer 203.0.113.2 liveness-detection multiplier 3
```

PE1: EVPN-MPLS Configuration

```
set interfaces lo0 unit 0 family inet address 198.51.100.1/32 primary
set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces xe-2/0/0 unit 0 family mpls
set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.111/24
set interfaces xe-2/0/2 unit 0 family mpls
set interfaces irb unit 1 family inet address 10.2.1.1/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.1/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 3 family inet address 10.2.3.1/24 virtual-gateway-address 10.2.3.254
set routing-options router-id 198.51.100.1
set routing-options autonomous-system 65000
set routing-options forwarding-table export evpn-pplb
set protocols mpls interface xe-2/0/0.0
set protocols mpls interface xe-2/0/2.0
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 198.51.100.1
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 198.51.100.2
set protocols bgp group evpn neighbor 198.51.100.3
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
set protocols ldp interface xe-2/0/0.0
set protocols ldp interface xe-2/0/2.0
set protocols ldp interface lo0.0
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface xe-2/0/6.1
set routing-instances evpn1 interface ae0.1
set routing-instances evpn1 interface ae1.1
set routing-instances evpn1 route-distinguisher 1:10
set routing-instances evpn1 vrf-target target:1:5
set routing-instances evpn1 protocols evpn extended-vlan-list 1
set routing-instances evpn1 protocols evpn mclag bgp-peer 198.51.100.2
set routing-instances evpn1 switch-options service-id 1
set routing-instances evpn1 vlans v1 vlan-id 1
set routing-instances evpn1 vlans v1 l3-interface irb.1
set routing-instances evpn1 vlans v1 no-arp-suppression
set routing-instances evpn2 instance-type virtual-switch
```

```
set routing-instances evpn2 interface xe-2/0/6.2
set routing-instances evpn2 interface ae0.2
set routing-instances evpn2 interface ae1.2
set routing-instances evpn2 route-distinguisher 1:20
set routing-instances evpn2 vrf-target target:1:6
set routing-instances evpn2 protocols evpn extended-vlan-list 2
set routing-instances evpn2 protocols evpn mclag bgp-peer 198.51.100.2
set routing-instances evpn2 switch-options service-id 2
set routing-instances evpn2 vlans v1 vlan-id 2
set routing-instances evpn2 vlans v1 l3-interface irb.2
set routing-instances evpn2 vlans v1 no-arp-suppression
set routing-instances evpn3 instance-type virtual-switch
set routing-instances evpn3 interface xe-2/0/6.3
set routing-instances evpn3 interface ae0.3
set routing-instances evpn3 interface ae1.3
set routing-instances evpn3 route-distinguisher 1:30
set routing-instances evpn3 vrf-target target:1:7
set routing-instances evpn3 protocols evpn extended-vlan-list 3
set routing-instances evpn3 protocols evpn mclag bgp-peer 198.51.100.2
set routing-instances evpn3 switch-options service-id 3
set routing-instances evpn3 vlans v1 vlan-id 3
set routing-instances evpn3 vlans v1 l3-interface irb.3
set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE2: MC-LAG Configuration

```
set chassis aggregated-devices ethernet device-count 3
set interfaces xe-2/0/1 gigether-options 802.3ad ae0
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.2/24
set interfaces xe-2/1/1 gigether-options 802.3ad ae1
set interfaces xe-2/1/2 gigether-options 802.3ad ae1
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
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 00:00:11:11:11:11
set interfaces ae0 aggregated-ether-options lacp admin-key 1
set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
set interfaces ae0 aggregated-ether-options mc-ae redundancy-group 2
set interfaces ae0 aggregated-ether-options mc-ae chassis-id 1
set interfaces ae0 aggregated-ether-options mc-ae mode active-active
set interfaces ae0 aggregated-ether-options mc-ae status-control standby
set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
set interfaces ae0 unit 1 esi all-active
set interfaces ae0 unit 1 family ethernet-switching interface-mode trunk
```

```
set interfaces ae0 unit 1 family ethernet-switching vlan members 1
set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 unit 2 esi all-active
set interfaces ae0 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 2 family ethernet-switching vlan members 2
set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
set interfaces ae0 unit 3 esi all-active
set interfaces ae0 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 3 family ethernet-switching vlan members 3
set interfaces ae1 flexible-vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 1 family ethernet-switching vlan members 1
set interfaces ae1 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 2 family ethernet-switching vlan members 2
set interfaces ae1 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 3 family ethernet-switching vlan members 3
set multi-chassis multi-chassis-protection 203.0.113.1 interface ae1
set protocols iccp local-ip-addr 203.0.113.2
set protocols iccp peer 203.0.113.1 session-establishment-hold-time 600
set protocols iccp peer 203.0.113.1 redundancy-group-id-list 2
set protocols iccp peer 203.0.113.1 liveness-detection minimum-interval 10000
set protocols iccp peer 203.0.113.1 liveness-detection multiplier 3
```

PE2: EVPN-MPLS Configuration

```
set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.222/24
set interfaces xe-2/0/0 unit 0 family mpls
set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.22/24
set interfaces xe-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.2/32 primary
set interfaces irb unit 1 family inet address 10.2.1.2/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.2/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 3 family inet address 10.2.3.2/24 virtual-gateway-address 10.2.3.254
set routing-options router-id 198.51.100.2
set routing-options autonomous-system 65000
set routing-options forwarding-table export evpn-pplb
set protocols mpls interface xe-2/0/2.0
set protocols mpls interface xe-2/0/0.0
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 198.51.100.2
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 198.51.100.1
set protocols bgp group evpn neighbor 198.51.100.3
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
set protocols ldp interface xe-2/0/0.0
set protocols ldp interface xe-2/0/2.0
set protocols ldp interface lo0.0
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
```

```
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface xe-2/0/6.1
set routing-instances evpn1 interface ae0.1
set routing-instances evpn1 interface ae1.1
set routing-instances evpn1 route-distinguisher 1:11
set routing-instances evpn1 vrf-target target:1:5
set routing-instances evpn1 protocols evpn extended-vlan-list 1
set routing-instances evpn1 protocols evpn mclag bgp-peer 198.51.100.1
set routing-instances evpn1 switch-options service-id 1
set routing-instances evpn1 vlans v1 vlan-id 1
set routing-instances evpn1 vlans v1 l3-interface irb.1
set routing-instances evpn1 vlans v1 no-arp-suppression
set routing-instances evpn2 instance-type virtual-switch
set routing-instances evpn2 interface xe-2/0/6.2
set routing-instances evpn2 interface ae0.2
set routing-instances evpn2 interface ae1.2
set routing-instances evpn2 route-distinguisher 1:21
set routing-instances evpn2 vrf-target target:1:6
set routing-instances evpn2 protocols evpn extended-vlan-list 2
set routing-instances evpn2 protocols evpn mclag bgp-peer 198.51.100.1
set routing-instances evpn2 switch-options service-id 2
set routing-instances evpn2 vlans v1 vlan-id 2
set routing-instances evpn2 vlans v1 l3-interface irb.2
set routing-instances evpn2 vlans v1 no-arp-suppression
set routing-instances evpn3 instance-type virtual-switch
set routing-instances evpn3 interface xe-2/0/6.3
set routing-instances evpn3 interface ae0.3
set routing-instances evpn3 interface ae1.3
set routing-instances evpn3 route-distinguisher 1:31
set routing-instances evpn3 vrf-target target:1:7
set routing-instances evpn3 protocols evpn extended-vlan-list 3
set routing-instances evpn3 protocols evpn mclag bgp-peer 198.51.100.1
set routing-instances evpn3 switch-options service-id 3
set routing-instances evpn3 vlans v1 vlan-id 3
set routing-instances evpn3 vlans v1 l3-interface irb.3
set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE1: Configuring MC-LAG

Step-by-Step Procedure

1. Set the number of aggregated Ethernet interfaces on PE1.

   ```
   [edit]
   user@switch# set chassis aggregated-devices ethernet device-count 3
   ```

2. Configure aggregated Ethernet interface ae0 on interface xe-2/0/1, and configure LACP and MC-LAG on ae0. Divide aggregated Ethernet interface ae0 into three logical interfaces (ae0.1, ae0.2, and ae0.3). For each logical interface, specify an ESI, place the logical interface is in MC-LAG active-active mode, and map the logical interface to a VLAN.

   ```
   [edit]
   user@switch# set interfaces xe-2/0/1 gigether-options 802.3ad ae0
   ```

```
user@switch# set interfaces ae0 flexible-vlan-tagging
user@switch# set interfaces ae0 encapsulation flexible-ethernet-services
user@switch# set interfaces ae0 aggregated-ether-options lacp active
user@switch# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@switch# set interfaces ae0 aggregated-ether-options lacp system-id
  00:00:11:11:11:11
user@switch# set interfaces ae0 aggregated-ether-options lacp admin-key 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae redundancy-group
  2
user@switch# set interfaces ae0 aggregated-ether-options mc-ae chassis-id 0
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mode
  active-active
user@switch# set interfaces ae0 aggregated-ether-options mc-ae status-control
  active
user@switch# set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set interfaces ae0 unit 1 esi all-active
user@switch# set interfaces ae0 unit 1 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae0 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
user@switch# set interfaces ae0 unit 2 esi all-active
user@switch# set interfaces ae0 unit 2 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae0 unit 2 family ethernet-switching vlan members 2
user@switch# set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
user@switch# set interfaces ae0 unit 3 esi all-active
user@switch# set interfaces ae0 unit 3 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae0 unit 3 family ethernet-switching vlan members 3
```

3. Configure physical interface xe-2/0/6, and divide it into three logical interfaces (xe-2/0/6.1, xe-2/0/6.2, and xe-2/0/6.3). Map each logical interface to a VLAN.

```
[edit]
user@switch# set interfaces xe-2/0/6 enable
user@switch# set interfaces xe-2/0/6 flexible-vlan-tagging
user@switch# set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members
  1
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members
  2
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members
  3
```

4. Configure physical interface xe-2/1/0 as a Layer 3 interface, on which you configure ICCP. Specify the interface with the IP address of 203.0.113.2 on PE2 as the ICCP peer to PE1.

```
[edit]
user@switch# set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.1/24
user@switch# set protocols iccp local-ip-addr 203.0.113.1
user@switch# set protocols iccp peer 203.0.113.2 session-establishment-hold-time
  600
user@switch# set protocols iccp peer 203.0.113.2 redundancy-group-id-list 2
user@switch# set protocols iccp peer 203.0.113.2 liveness-detection
  minimum-interval 10000
user@switch# set protocols iccp peer 203.0.113.2 liveness-detection multiplier 3
```

5. Configure aggregated Ethernet interface ae1 on interfaces xe-2/1/1 and xe-2/1/2, and configure LACP on ae1. Divide aggregated Ethernet interface ae1 into three logical interfaces (ae1.1, ae1.2, and ae1.3), and map each logical interface to a VLAN. Specify ae1 as the multichassis protection link between PE1 and PE2.

```
[edit]
user@switch# set interfaces xe-2/1/1 gigether-options 802.3ad ae1
user@switch# set interfaces xe-2/1/2 gigether-options 802.3ad ae1
user@switch# set interfaces ae1 flexible-vlan-tagging
user@switch# set interfaces ae1 encapsulation flexible-ethernet-services
user@switch# set interfaces ae1 aggregated-ether-options lacp active
user@switch# set interfaces ae1 unit 1 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae1 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces ae1 unit 2 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae1 unit 2 family ethernet-switching vlan members 2
user@switch# set interfaces ae1 unit 3 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae1 unit 3 family ethernet-switching vlan members 3
user@switch# set multi-chassis multi-chassis-protection 203.0.113.2 interface ae1
```

PE1: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, and the interfaces connected to the other PE devices.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 198.51.100.1/32 primary
user@switch# set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.2/24
user@switch# set interfaces xe-2/0/0 unit 0 family mpls
user@switch# set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.111/24
user@switch# set interfaces xe-2/0/2 unit 0 family mpls
```

2. Configure IRB interfaces irb.1, irb.2, and irb.3.

```
[edit]
user@switch# set interfaces irb unit 1 family inet address 10.2.1.1/24
  virtual-gateway-address 10.2.1.254
user@switch# set interfaces irb unit 2 family inet address 10.2.2.1/24
  virtual-gateway-address 10.2.2.254
user@switch# set interfaces irb unit 3 family inet address 10.2.3.1/24
  virtual-gateway-address 10.2.3.254
```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 198.51.100.1
user@switch# set routing-options autonomous-system 65000
```

4. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```
[edit]
user@switch# set routing-options forwarding-table export evpn-pplb
user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
user@switch# set policy-options policy-statement evpn-pplb then load-balance
  per-packet
```

5. Enable MPLS on interfaces xe-2/0/0.0 and xe-2/0/2.0.

```
[edit]
user@switch# set protocols mpls interface xe-2/0/0.0
user@switch# set protocols mpls interface xe-2/0/2.0
```

6. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
user@switch# set protocols bgp group evpn type internal
user@switch# set protocols bgp group evpn local-address 198.51.100.1
user@switch# set protocols bgp group evpn family evpn signaling
user@switch# set protocols bgp group evpn local-as 65000
user@switch# set protocols bgp group evpn neighbor 198.51.100.2
user@switch# set protocols bgp group evpn neighbor 198.51.100.3
```

7. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
```

```
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
```

8. Configure the Label Distribution Protocol (LDP) on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface xe-2/0/0.0
user@switch# set protocols ldp interface xe-2/0/2.0
```

9. Configure virtual switch routing instances for VLAN v1, which is assigned VLAN IDs of 1, 2, and 3, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface xe-2/0/6.1
user@switch# set routing-instances evpn1 interface ae0.1
user@switch# set routing-instances evpn1 interface ae1.1
user@switch# set routing-instances evpn1 route-distinguisher 1:10
user@switch# set routing-instances evpn1 vrf-target target:1:5
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 1
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer
  198.51.100.2
user@switch# set routing-instances evpn1 switch-options service-id 1
user@switch# set routing-instances evpn1 vlans v1 vlan-id 1
user@switch# set routing-instances evpn1 vlans v1 l3-interface irb.1
user@switch# set routing-instances evpn1 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn2 instance-type virtual-switch
user@switch# set routing-instances evpn2 interface xe-2/0/6.2
user@switch# set routing-instances evpn2 interface ae0.2
user@switch# set routing-instances evpn2 interface ae1.2
user@switch# set routing-instances evpn2 route-distinguisher 1:20
user@switch# set routing-instances evpn2 vrf-target target:1:6
user@switch# set routing-instances evpn2 protocols evpn extended-vlan-list 2
user@switch# set routing-instances evpn2 protocols evpn mclag bgp-peer
  198.51.100.2
user@switch# set routing-instances evpn2 switch-options service-id 2
user@switch# set routing-instances evpn2 vlans v1 vlan-id 2
user@switch# set routing-instances evpn2 vlans v1 l3-interface irb.2
user@switch# set routing-instances evpn2 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn3 instance-type virtual-switch
user@switch# set routing-instances evpn3 interface xe-2/0/6.3
user@switch# set routing-instances evpn3 interface ae0.3
user@switch# set routing-instances evpn3 interface ae1.3
user@switch# set routing-instances evpn3 route-distinguisher 1:30
user@switch# set routing-instances evpn3 vrf-target target:1:7
user@switch# set routing-instances evpn3 protocols evpn extended-vlan-list 3
user@switch# set routing-instances evpn3 protocols evpn mclag bgp-peer
  198.51.100.2
user@switch# set routing-instances evpn3 switch-options service-id 3
user@switch# set routing-instances evpn3 vlans v1 vlan-id 3
user@switch# set routing-instances evpn3 vlans v1 l3-interface irb.3
```

```
user@switch# set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE2: Configuring MC-LAG

Step-by-Step Procedure

1. Set the number of aggregated Ethernet interfaces on PE2.

```
[edit]
user@switch# set chassis aggregated-devices ethernet device-count 3
```

2. Configure aggregated Ethernet interface ae0 on interface xe-2/0/1, and configure LACP and MC-LAG on ae0. Divide aggregated Ethernet interface ae0 into three logical interfaces (ae0.1, ae0.2, and ae0.3). For each logical interface, specify an ESI, place the logical interface is in MC-LAG active-active mode, and map the logical interface to a VLAN.

```
[edit]
user@switch# set interfaces xe-2/0/1 gigether-options 802.3ad ae0
user@switch# set interfaces ae0 flexible-vlan-tagging
user@switch# set interfaces ae0 encapsulation flexible-ethernet-services
user@switch# set interfaces ae0 aggregated-ether-options lacp active
user@switch# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@switch# set interfaces ae0 aggregated-ether-options lacp system-id
  00:00:11:11:11:11
user@switch# set interfaces ae0 aggregated-ether-options lacp admin-key 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mc-ae-id 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae redundancy-group
  2
user@switch# set interfaces ae0 aggregated-ether-options mc-ae chassis-id 1
user@switch# set interfaces ae0 aggregated-ether-options mc-ae mode
  active-active
user@switch# set interfaces ae0 aggregated-ether-options mc-ae status-control
  standby
user@switch# set interfaces ae0 unit 1 esi 00:11:22:33:44:55:66:77:88:99
user@switch# set interfaces ae0 unit 1 esi all-active
user@switch# set interfaces ae0 unit 1 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae0 unit 1 family ethernet-switching vlan members 1
user@switch# set interfaces ae0 unit 2 esi 00:11:11:11:11:11:11:11:11:11
user@switch# set interfaces ae0 unit 2 esi all-active
user@switch# set interfaces ae0 unit 2 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae0 unit 2 family ethernet-switching vlan members 2
user@switch# set interfaces ae0 unit 3 esi 00:11:22:22:22:22:22:22:22:22
user@switch# set interfaces ae0 unit 3 esi all-active
user@switch# set interfaces ae0 unit 3 family ethernet-switching interface-mode
  trunk
user@switch# set interfaces ae0 unit 3 family ethernet-switching vlan members 3
```

3. Configure physical interface xe-2/0/6, and divide it into three logical interfaces (xe-2/0/6.1, xe-2/0/6.2, and xe-2/0/6.3). Map each logical interface to a VLAN.

```
[edit]
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
```

4. Configure physical interface xe-2/1/0 as a Layer 3 interface, on which you configure ICCP. Specify the interface with the IP address of 203.0.113.1 on PE1 as the ICCP peer to PE2.

```
[edit]
set interfaces xe-2/1/0 unit 0 family inet address 203.0.113.2/24
set protocols iccp local-ip-addr 203.0.113.2
set protocols iccp peer 203.0.113.1 session-establishment-hold-time 600
set protocols iccp peer 203.0.113.1 redundancy-group-id-list 2
set protocols iccp peer 203.0.113.1 liveness-detection minimum-interval 10000
set protocols iccp peer 203.0.113.1 liveness-detection multiplier 3
```

5. Configure aggregated Ethernet interface ae1 on interfaces xe-2/1/1 and xe-2/1/2, and configure LACP on ae1. Divide aggregated Ethernet interface ae1 into three logical interfaces (ae1.1, ae1.2, and ae1.3), and map each logical interface to a VLAN. Specify ae1 as the multichassis protection link between PE1 and PE2.

```
[edit]
set interfaces xe-2/1/1 gigether-options 802.3ad ae1
set interfaces xe-2/1/2 gigether-options 802.3ad ae1
set interfaces ae1 flexible-vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 aggregated-ether-options lacp active
set interfaces ae1 unit 1 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 1 family ethernet-switching vlan members 1
set interfaces ae1 unit 2 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 2 family ethernet-switching vlan members 2
set interfaces ae1 unit 3 family ethernet-switching interface-mode trunk
set interfaces ae1 unit 3 family ethernet-switching vlan members 3
set multi-chassis multi-chassis-protection 203.0.113.1 interface ae1
```

PE2: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, and the interfaces connected to the other PE devices.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 198.51.100.2/32 primary
user@switch# set interfaces xe-2/0/0 unit 0 family inet address 192.0.2.222/24
user@switch# set interfaces xe-2/0/0 unit 0 family mpls
user@switch# set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.22/24
user@switch# set interfaces xe-2/0/2 unit 0 family mpls
```

2. Configure IRB interfaces irb.1, irb.2, and irb.3.

```
[edit]
user@switch# set interfaces irb unit 1 family inet address 10.2.1.2/24
  virtual-gateway-address 10.2.1.254
user@switch# set interfaces irb unit 2 family inet address 10.2.2.2/24
  virtual-gateway-address 10.2.2.254
user@switch# set interfaces irb unit 3 family inet address 10.2.3.2/24
  virtual-gateway-address 10.2.3.254
```

3. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 198.51.100.2
user@switch# set routing-options autonomous-system 65000
```

4. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```
[edit]
user@switch# set routing-options forwarding-table export evpn-pplb
user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
user@switch# set policy-options policy-statement evpn-pplb then load-balance
  per-packet
```

5. Enable MPLS on interfaces xe-2/0/0.0 and xe-2/0/2.0.

```
[edit]
user@switch# set protocols mpls interface xe-2/0/0.0
user@switch# set protocols mpls interface xe-2/0/2.0
```

6. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
```

```
user@switch# set protocols bgp group evpn type internal
user@switch# set protocols bgp group evpn local-address 198.51.100.2
user@switch# set protocols bgp group evpn family evpn signaling
user@switch# set protocols bgp group evpn local-as 65000
user@switch# set protocols bgp group evpn neighbor 198.51.100.1
user@switch# set protocols bgp group evpn neighbor 198.51.100.3
```

7. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
```

8. Configure the Label Distribution Protocol (LDP) on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface xe-2/0/0.0
user@switch# set protocols ldp interface xe-2/0/2.0
```

9. Configure virtual switch routing instances for VLAN v1, which is assigned VLAN IDs of 1, 2, and 3, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface xe-2/0/6.1
user@switch# set routing-instances evpn1 interface ae0.1
user@switch# set routing-instances evpn1 interface ae1.1
user@switch# set routing-instances evpn1 route-distinguisher 1:11
user@switch# set routing-instances evpn1 vrf-target target:1:5
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 1
user@switch# set routing-instances evpn1 protocols evpn mclag bgp-peer
  198.51.100.1
user@switch# set routing-instances evpn1 switch-options service-id 1
user@switch# set routing-instances evpn1 vlans v1 vlan-id 1
user@switch# set routing-instances evpn1 vlans v1 l3-interface irb.1
user@switch# set routing-instances evpn1 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn2 instance-type virtual-switch
user@switch# set routing-instances evpn2 interface xe-2/0/6.2
user@switch# set routing-instances evpn2 interface ae0.2
user@switch# set routing-instances evpn2 interface ae1.2
user@switch# set routing-instances evpn2 route-distinguisher 1:21
user@switch# set routing-instances evpn2 vrf-target target:1:6
user@switch# set routing-instances evpn2 protocols evpn extended-vlan-list 2
user@switch# set routing-instances evpn2 protocols evpn mclag bgp-peer
  198.51.100.1
user@switch# set routing-instances evpn2 switch-options service-id 2
```

```
user@switch# set routing-instances evpn2 vlans v1 vlan-id 2
user@switch# set routing-instances evpn2 vlans v1 l3-interface irb.2
user@switch# set routing-instances evpn2 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn3 instance-type virtual-switch
user@switch# set routing-instances evpn3 interface xe-2/0/6.3
user@switch# set routing-instances evpn3 interface ae0.3
user@switch# set routing-instances evpn3 interface ae1.3
user@switch# set routing-instances evpn3 route-distinguisher 1:31
user@switch# set routing-instances evpn3 vrf-target target:1:7
user@switch# set routing-instances evpn3 protocols evpn extended-vlan-list 3
user@switch# set routing-instances evpn3 protocols evpn mclag bgp-peer
  198.51.100.1
user@switch# set routing-instances evpn3 switch-options service-id 3
user@switch# set routing-instances evpn3 vlans v1 vlan-id 3
user@switch# set routing-instances evpn3 vlans v1 l3-interface irb.3
user@switch# set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE3 Configuration

CLI Quick Configuration

PE3: EVPN-MPLS Configuration

```
set interfaces lo0 unit 0 family inet address 198.51.100.3/32 primary
set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.1/24
set interfaces xe-2/0/2 unit 0 family mpls
set interfaces xe-2/0/3 unit 0 family inet address 192.0.2.11/24
set interfaces xe-2/0/3 unit 0 family mpls
set interfaces xe-2/0/6 enable
set interfaces xe-2/0/6 flexible-vlan-tagging
set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
set interfaces xe-2/0/6 unit 1 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members 1
set interfaces xe-2/0/6 unit 2 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members 2
set interfaces xe-2/0/6 unit 3 family ethernet-switching interface-mode trunk
set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members 3
set interfaces irb unit 1 family inet address 10.2.1.3/24 virtual-gateway-address 10.2.1.254
set interfaces irb unit 2 family inet address 10.2.2.3/24 virtual-gateway-address 10.2.2.254
set interfaces irb unit 3 family inet address 10.2.3.3/24 virtual-gateway-address 10.2.3.254
set routing-options router-id 198.51.100.3
set routing-options autonomous-system 65000
set routing-options forwarding-table export evpn-pplb
set protocols mpls interface xe-2/0/2.0
set protocols mpls interface xe-2/0/3.0
set protocols bgp group evpn type internal
set protocols bgp group evpn local-address 198.51.100.3
set protocols bgp group evpn family evpn signaling
set protocols bgp group evpn local-as 65000
set protocols bgp group evpn neighbor 198.51.100.1
set protocols bgp group evpn neighbor 198.51.100.2
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
set protocols ospf area 0.0.0.0 interface xe-2/0/3.0
```

```
set protocols ldp interface lo0.0
set protocols ldp interface xe-2/0/2.0
set protocols ldp interface xe-2/0/3.0
set policy-options policy-statement evpn-pplb from protocol evpn
set policy-options policy-statement evpn-pplb then load-balance per-packet
set routing-instances evpn1 instance-type virtual-switch
set routing-instances evpn1 interface xe-2/0/6.1
set routing-instances evpn1 route-distinguisher 1:12
set routing-instances evpn1 vrf-target target:1:5
set routing-instances evpn1 protocols evpn extended-vlan-list 1
set routing-instances evpn1 switch-options service-id 1
set routing-instances evpn1 vlans v1 vlan-id 1
set routing-instances evpn1 vlans v1 l3-interface irb.1
set routing-instances evpn1 vlans v1 no-arp-suppression
set routing-instances evpn2 instance-type virtual-switch
set routing-instances evpn2 interface xe-2/0/6.2
set routing-instances evpn2 route-distinguisher 1:22
set routing-instances evpn2 vrf-target target:1:6
set routing-instances evpn2 protocols evpn extended-vlan-list 2
set routing-instances evpn2 switch-options service-id 2
set routing-instances evpn2 vlans v1 vlan-id 2
set routing-instances evpn2 vlans v1 l3-interface irb.2
set routing-instances evpn2 vlans v1 no-arp-suppression
set routing-instances evpn3 instance-type virtual-switch
set routing-instances evpn3 interface xe-2/0/6.3
set routing-instances evpn3 route-distinguisher 1:32
set routing-instances evpn3 vrf-target target:1:7
set routing-instances evpn3 protocols evpn extended-vlan-list 3
set routing-instances evpn3 switch-options service-id 3
set routing-instances evpn3 vlans v1 vlan-id 3
set routing-instances evpn3 vlans v1 l3-interface irb.3
set routing-instances evpn3 vlans v1 no-arp-suppression
```

PE3: Configuring EVPN-MPLS

Step-by-Step Procedure

1. Configure the loopback interface, and the interfaces connected to the other PE devices.

```
[edit]
user@switch# set interfaces lo0 unit 0 family inet address 198.51.100.3/32 primary
user@switch# set interfaces xe-2/0/2 unit 0 family inet address 192.0.2.1/24
user@switch# set interfaces xe-2/0/2 unit 0 family mpls
user@switch# set interfaces xe-2/0/3 unit 0 family inet address 192.0.2.11/24
user@switch# set interfaces xe-2/0/3 unit 0 family mpls
```

2. Configure interface xe-2/0/6, which is connected to the host.

```
[edit]
user@switch# set interfaces xe-2/0/6 enable
user@switch# set interfaces xe-2/0/6 flexible-vlan-tagging
user@switch# set interfaces xe-2/0/6 encapsulation flexible-ethernet-services
```

```
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 1 family ethernet-switching vlan members
  1
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 2 family ethernet-switching vlan members
  2
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching
  interface-mode trunk
user@switch# set interfaces xe-2/0/6 unit 3 family ethernet-switching vlan members
  3
```

3. Configure IRB interfaces irb.1, irb.2, and irb.3.

```
[edit]
user@switch# set interfaces irb unit 1 family inet address 10.2.1.3/24
  virtual-gateway-address 10.2.1.254
user@switch# set interfaces irb unit 2 family inet address 10.2.2.3/24
  virtual-gateway-address 10.2.2.254
user@switch# set interfaces irb unit 3 family inet address 10.2.3.3/24
  virtual-gateway-address 10.2.3.254
```

4. Assign a router ID and the autonomous system in which PE1, PE2, and PE3 reside.

```
[edit]
user@switch# set routing-options router-id 198.51.100.3
user@switch# set routing-options autonomous-system 65000
```

5. Enable per-packet load-balancing for EVPN routes when EVPN multihoming active-active mode is used.

```
[edit]
user@switch# set routing-options forwarding-table export evpn-pplb
user@switch# set policy-options policy-statement evpn-pplb from protocol evpn
user@switch# set policy-options policy-statement evpn-pplb then load-balance
  per-packet
```

6. Enable MPLS on interfaces xe-2/0/2.0 and xe-2/0/3.0.

```
[edit]
user@switch# set protocols mpls interface xe-2/0/2.0
user@switch# set protocols mpls interface xe-2/0/3.0
```

7. Configure an IBGP overlay that includes PE1, PE2, and PE3.

```
[edit]
```

```
user@switch# set protocols bgp group evpn type internal
user@switch# set protocols bgp group evpn local-address 198.51.100.3
user@switch# set protocols bgp group evpn family evpn signaling
user@switch# set protocols bgp group evpn local-as 65000
user@switch# set protocols bgp group evpn neighbor 198.51.100.1
user@switch# set protocols bgp group evpn neighbor 198.51.100.2
```

8. Configure OSPF as the internal routing protocol for EVPN by specifying an area ID and interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ospf area 0.0.0.0 interface lo0.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/2.0
user@switch# set protocols ospf area 0.0.0.0 interface xe-2/0/3.0
```

9. Configure the LDP on the loopback interface and the interfaces on which EVPN-MPLS is enabled.

```
[edit]
user@switch# set protocols ldp interface lo0.0
user@switch# set protocols ldp interface xe-2/0/2.0
user@switch# set protocols ldp interface xe-2/0/3.0
```

10. Configure virtual switch routing instances for VLAN v1, which is assigned VLAN IDs of 1, 2, and 3, and include the interfaces and other entities associated with the VLAN.

```
[edit]
user@switch# set routing-instances evpn1 instance-type virtual-switch
user@switch# set routing-instances evpn1 interface xe-2/0/6.1
user@switch# set routing-instances evpn1 route-distinguisher 1:12
user@switch# set routing-instances evpn1 vrf-target target:1:5
user@switch# set routing-instances evpn1 protocols evpn extended-vlan-list 1
user@switch# set routing-instances evpn1 switch-options service-id 1
user@switch# set routing-instances evpn1 vlans v1 vlan-id 1
user@switch# set routing-instances evpn1 vlans v1 l3-interface irb.1
user@switch# set routing-instances evpn1 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn2 instance-type virtual-switch
user@switch# set routing-instances evpn2 interface xe-2/0/6.2
user@switch# set routing-instances evpn2 route-distinguisher 1:22
user@switch# set routing-instances evpn2 vrf-target target:1:6
user@switch# set routing-instances evpn2 protocols evpn extended-vlan-list 2
user@switch# set routing-instances evpn2 switch-options service-id 2
user@switch# set routing-instances evpn2 vlans v1 vlan-id 2
user@switch# set routing-instances evpn2 vlans v1 l3-interface irb.2
user@switch# set routing-instances evpn2 vlans v1 no-arp-suppression
user@switch# set routing-instances evpn3 instance-type virtual-switch
user@switch# set routing-instances evpn3 interface xe-2/0/6.3
user@switch# set routing-instances evpn3 route-distinguisher 1:32
user@switch# set routing-instances evpn3 vrf-target target:1:7
user@switch# set routing-instances evpn3 protocols evpn extended-vlan-list 3
```

```
user@switch# set routing-instances evpn3 switch-options service-id 3
user@switch# set routing-instances evpn3 vlans v1 vlan-id 3
user@switch# set routing-instances evpn3 vlans v1 l3-interface irb.3
user@switch# set routing-instances evpn3 vlans v1 no-arp-suppression
```

PART 7

PBB-EVPN

CHAPTER 27

Configuring PBB-EVPN Integration

Provider Backbone Bridging (PBB) and EVPN Integration Overview

Ethernet VPN (EVPN) provides a solution for multipoint Layer 2 VPN services with advanced multihoming capabilities using BGP for distributing MAC address reachability information over the core MPLS or IP network. However, with EVPN, several thousands of MAC addresses are carried from each virtual routing and forwarding (VRF) instance, requiring frequent updates on newly learned MAC routes and withdrawn routes. This increases the overhead on the provider network.

Provider backbone bridging (PBB) extends Layer 2 Ethernet switching to provide enhanced scalability, quality-of-service (QoS) features, and carrier-class reliability. With the integration of PBB with EVPN, instead of sending the customer MAC (C-MAC) addresses as control plane learning, the backbone MAC (B-MAC) addresses are distributed in the EVPN core. This simplifies the control plane learning across the core and allows a huge number of Layer 2 services, such as data center connectivity, to transit the network in a simple manner.

The following sections describe the technology and implementation overview of PBB-EVPN integration:

Technology Overview of PBB-EVPN Integration

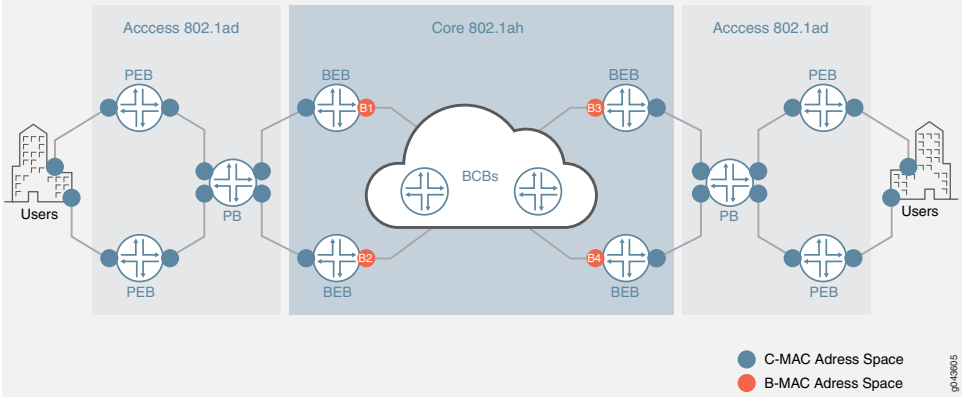
Understanding Provider Backbone Bridging (PBB)

Provider backbone bridging (PBB) was originally defined as IEEE 802.1ah standard, and operates in exactly the same manner as IEEE 802.1ad standard. However, instead of multiplexing VLANs, PBB duplicates the MAC layer of the customer frame and separates it from the provider domain, by encapsulating it in a 24 bit instance service identifier (I-SID). This allows for complete transparency between the customer network and the provider network.

When operating on customer MAC (C-MAC) and service MAC (S-MAC) addresses, PBB uses a new backbone MAC (B-MAC) address. The B-MAC address is added at the edge of the PBB network, that is administered by a carrier VPN or a carrier-of-carriers VPN. With the use of an I-SID for the customer routing instance (I-component) service group, PBB improves the scalability of Ethernet services.

Figure 91 on page 954 illustrates a PBB network, describing the PBB network elements and MAC address spaces.

Figure 91: PBB Network Elements



The PBB terms are:

- **PB**—Provider bridge (802.1ad)
- **PEB**—Provider edge bridge (802.1ad)
- **BEB**—Backbone edge bridge (802.1ah)
- **BCB**—Backbone core bridge (802.1ah)

The BEB device is the first immediate point of interest within PBB, and forms the boundary between the access network and the core. This introduces two key components—the I-component and B-component in PBB.

- **I-component**

  The I-component forms the customer or access facing interface or routing instance. The I-component is responsible for mapping customer Ethernet traffic to the appropriate I-SID. At first, the customer Ethernet traffic is mapped to a customer bridge

domain. Then each customer bridge domain is mapped to an I-SID. This service mapping can be per port, per port with service VLAN (S-VLAN), or per port with S-VLAN and customer VLAN (C-VLAN). The I-component is used to learn and forward frames based on the C-MAC addresses, and maintains a C-MAC-to-B-MAC mapping table that is based on instance tag (I-TAG).

Within the I-component there are two ports:

- Customer Instance Port (CIP)

  These ports are customer service instances at the customer-facing interfaces. Service definitions can be per port, per port with S-VLAN, or per port with S-VLAN and C-VLAN.

- Provider Instance Port (PIP)

  This port performs PBB encapsulation, such as pushing the I-TAG, source and destination B-MAC addresses, and PBB decapsulation, such as popping the I-SID, learning source B-MAC-to-C-MAC mapping, in the ingress direction.

- **B-component**

  the B-component is the backbone-facing PBB core instance. The B-component is used to learn and forward packets based on the B-MAC addresses. The B-component is then responsible for mapping the I-SIDs to the appropriate B-VLANs (in the case of PBB networks) or pushing and popping service MPLS labels for MPLS-based networks.

  Within the B-component there are two ports:

  - Customer Backbone Port (CBP)

    These ports are backbone edge ports that can receive and transmit instance-tagged frames from multiple customers, and assign backbone VLAN IDs (B-VIDs) and translate the I-SID on the basis of the received I-SID.

  - Provider Backbone Port (PBP)

    These ports provide connectivity to the other bridges within and attached to the backbone. These are provider-facing ports. These ports support the S-VLAN component.

Figure 92 on page 956 illustrates the key components of PBB. Figure 93 on page 956 illustrates the PBB packet format.

Figure 92: PBB Key Components

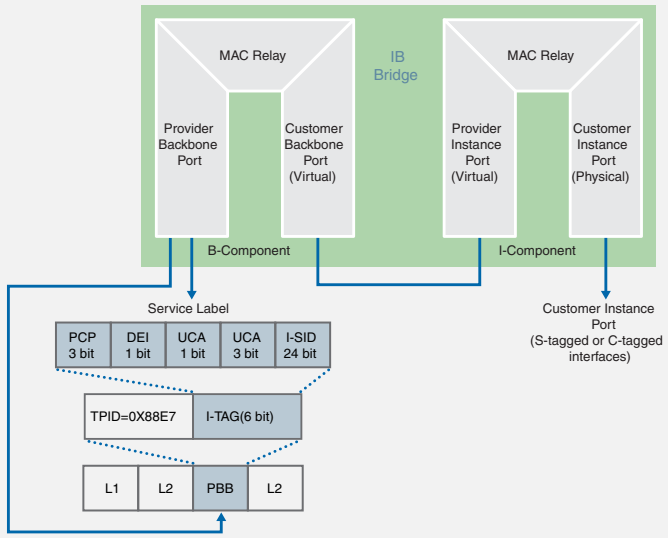
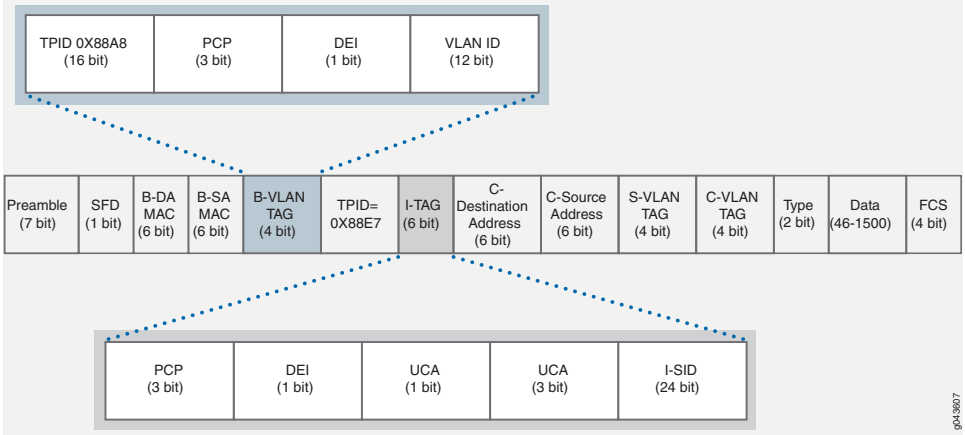


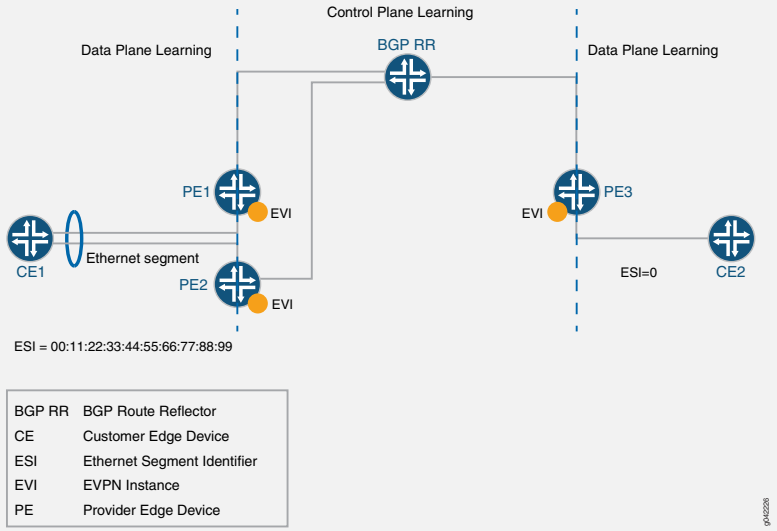
Figure 93: PBB Packet Format



Understanding EVPN

EVPN is a new standards-based technology that provides virtual multipoint bridged connectivity between different Layer 2 domains over an IP or IP/MPLS backbone network. Similar to other VPN technologies, such as IPVPN and VPLS, EVPN instances (EVIs) are configured on PE routers to maintain logical service separation between customers. The provider edge (PE) devices connect to customer edge (CE) devices, which can be a router, switch, or host. The PE devices then exchange reachability information using MultiProtocol BGP (MP-BGP) and encapsulated traffic is forwarded between them. Because elements of the architecture are common with other VPN technologies, EVPN can be seamlessly introduced and integrated into existing service environments.

Figure 94: EVPN Overview



The EVPN technology provides mechanisms for next-generation DCI by adding extended control plane procedures to exchange the Layer 2 (MAC address) and Layer 3 (IP address) information among the participating Data Center Border Routers (DCBRs). These features help address some of the DCI challenges, such as seamless VM mobility and optimal IP routing. Seamless VM mobility refers to the challenge of Layer 2 extension and maintaining connectivity in the face of VM mobility, and optimal IP routing refers to the challenge of supporting default gateway behavior for a VM's outbound traffic and triangular routing avoidance of a VM's inbound traffic.

The EVPN technology is used by the data center operator to offer multi-tenancy, flexible, and resilient services that can be extended on demand. This flexibility and resiliency can require using compute resources among different physical data centers for a single service (Layer 2 extension) and VM motion.

EVPN supports all-active multihoming, which allows a CE device to connect to two or more PE devices such that traffic is forwarded using all of the links between the devices. This enables the CE device to load-balance traffic to the multiple PE devices. More importantly it allows a remote PE device to load-balance traffic to the multihomed PEs across the core network. This load balancing of traffic flows between data centers is known as aliasing. EVPN also has mechanisms that prevent the looping of broadcast, unknown unicast, and multicast (BUM) traffic in an all-active multihomed topology.

Multihoming provides redundancy in the event that an access link or a PE device fails. In either case, traffic flows from the CE device toward the PE device use the remaining active links. For traffic in the other direction, the remote PE device updates its forwarding table to send traffic to the remaining active PE devices connected to the multihomed Ethernet segment. EVPN provides a fast convergence mechanism so that the time it takes to make this adjustment is independent of the number of MAC addresses learned by the PE device.

EVPN's MP-BGP control plane allows live virtual machines to be dynamically moved from one data center to another, also known as VM motion. After a VM is moved to a destination server/hypervisor, it transmits a gratuitous ARP that updates the Layer 2 forwarding table of the PE device at the destination data center. The PE device then transmits a MAC route update to all remote PE devices which in turn update their forwarding tables. In this manner, an EVPN tracks the movement of the VM, also known as MAC Mobility. EVPN also has mechanisms to detect and stop MAC flapping.

The EVPN technology, similar to Layer 3 MPLS VPN, introduces the concept of routing MAC addresses using MP-BGP over the MPLS core. Some of the important benefits of using EVPNs include:

- Ability to have a dual-active multihomed edge device
- Provides load balancing across dual-active links
- Provides MAC address mobility
- Provides multi-tenancy
- Provides aliasing
- Enables fast convergence

PBB-EVPN Integration

The integration of PBB with EVPN is described in the following sections:

- Integrating the PBB and EVPN Network Elements on page 958
- PBB-EVPN Control Plane Initialization on page 959
- Discovery of EVPN Routes in PBB-EVPN on page 960

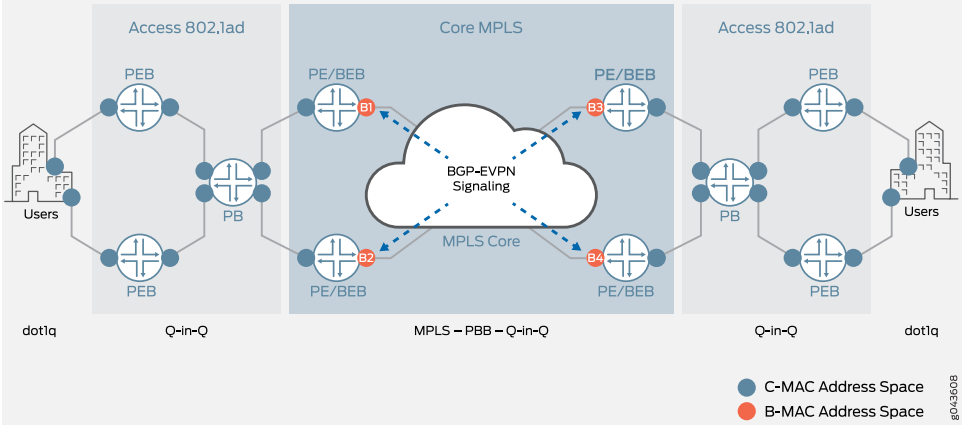
Integrating the PBB and EVPN Network Elements

In a PBB network, a huge amount of customer MAC (C-MAC) addresses are hid behind a drastically smaller number of backbone MAC (B-MAC) addresses, without the devices in the core having to learn and process all the individual customer states. The I-SID creates an encapsulation that enables a large number of services to be deployed. However, unlike modern networks that have a simple MPLS core composed of PE and provider devices, the devices in the PBB core need to act as switches, called the backbone core bridge (BCB), performing forwarding decisions based on B-MAC addresses. This causes incompatibility issues with modern MPLS networks, where packets are switched between edge loopback addresses using MPLS labels and recursion.

With the integration of PBB with EVPN, the BCB element in the PBB core is replaced with MPLS, while retaining the service scaling properties of the BEB PBB edge device. The B-component is signaled using EVPN BGP signaling and encapsulated inside MPLS using PE and provider devices. As a result, the vast scale of PBB is combined with the simplicity of a traditional basic MPLS core network and the amount of network-wide state information is significantly reduced, as opposed to regular PBB.

Figure 95 on page 959 illustrates the PBB-EVPN integration using the different elements of a PBB and EVPN network.

Figure 95: PBB-EVPN Integration

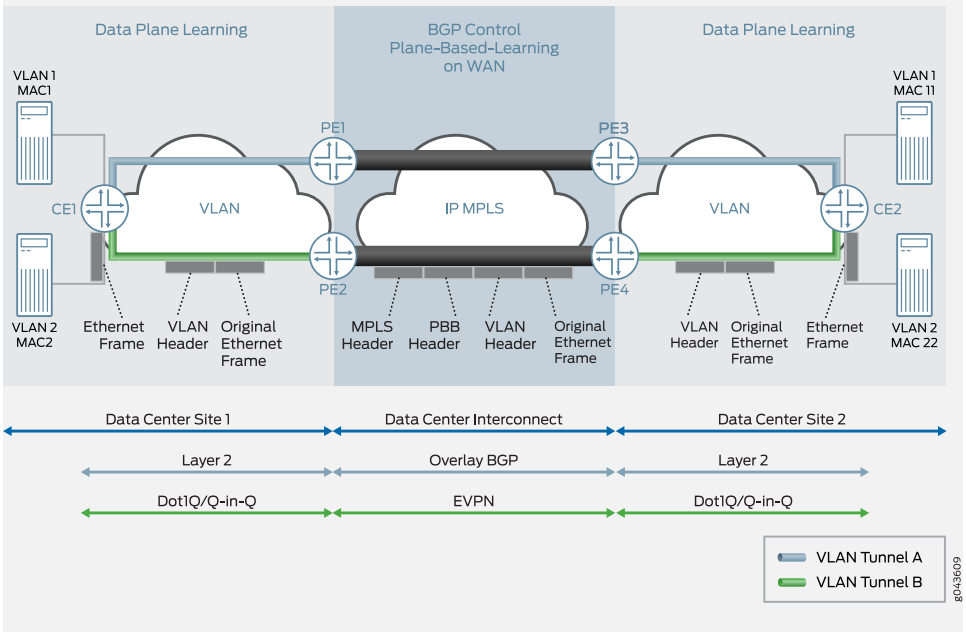


PBB-EVPN Control Plane Initialization

In a PBB-EVPN network, B-MAC addresses are distributed over the EVPN core and C-MAC addresses are learned in the data plane and aggregated behind the B-MAC addresses.

Figure 96 on page 959 illustrates the control plane handling in a sample PBB-EVPN network with four PE devices and two top-of-rack devices across two data centers.

Figure 96: PBB-EVPN Control Plane Handling



The control plane handling at the Data Center Site 1 is as follows:

1. The C-MAC address lookup occurs and the C-MAC address is learned.
2. The B-MAC source address and I-SID are pushed on the packet.
3. Destination address lookup of C-MAC to B-MAC is done in the I-SID table. If the MAC address is present, the packet is routed using an EVPN MAC route; otherwise a multicast route is used.
4. This route gives the service label for the packet, which has PBB and also the original frame.

The control plane handling at the Data Center Site 2 is as follows:

1. At the disposition PE device, the packet is received with a single service label, indicating that it is a PBB frame.
2. The source address allocation of C-MAC to B-MAC is learned in the I-SID table.
3. The source address of the C-MAC is learned in the customer bridge domain (C-BD) MAC table.

Discovery of EVPN Routes in PBB-EVPN

PBB with Dot1ah functionality is implemented at the PE devices. In the case of PBB-EVPN, the PE devices implement the instance and backbone bridge functionality. Only B-MAC addresses are distributed in the control plane, the C-MAC addresses are learned in the data plane. The following EVPN routes are discovered on the different PE devices:

- VPN Autodiscovery on page 960
- Ethernet Segment Discovery on page 961
- Ethernet MAC Routes Discovery on page 962
- Differences Between PBB-EVPN and EVPN on page 963

VPN Autodiscovery

When an EVPN instance (EVI) is configured on different PE devices, autodiscovery of the VPN happens first for discovering the EVPN endpoints. Each PE device that is configured with an EVI sends the inclusive multicast route.

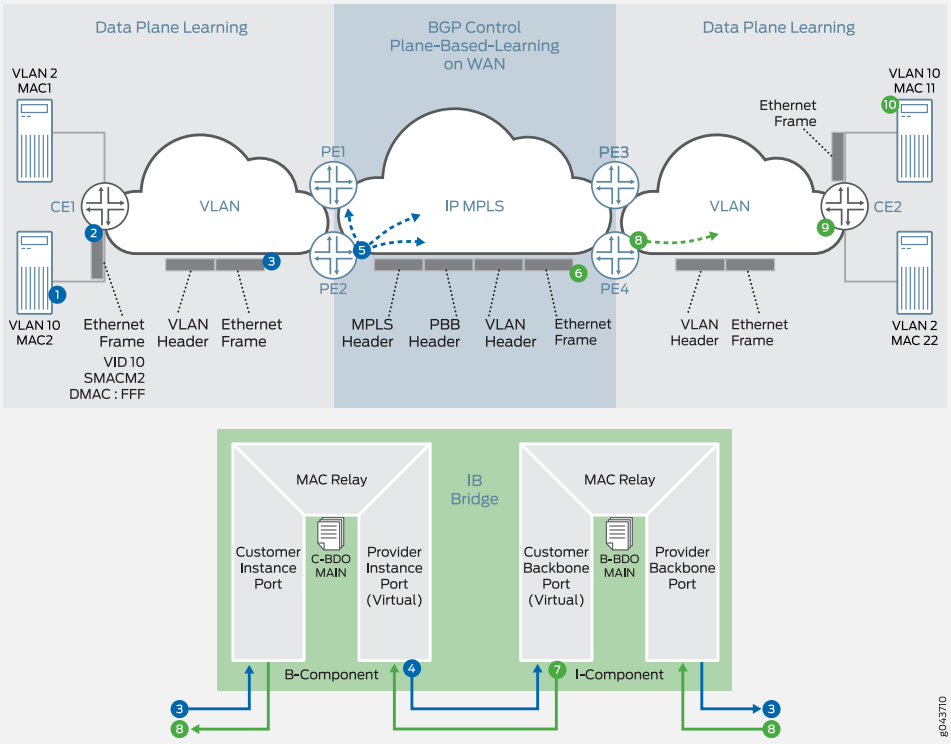
The inclusive multicast route fields are as follows:

- **RD**—Unique route distinguisher value per advertising PE device per EVI. The significance of the RD is local to a PE device.
- **TAG ID**—Service ID equivalent to the I-SID value. One I-SID is assigned to one bridge domain under an EVI when service is supported. The TAG ID is set to 0 for the I-SID bundle service, where multiple I-SIDs are mapped to one bridge domain.

- **Originating IP addr**—Loopback IP address.
- **P-Multicast Service Interface (PMSI) Attributes**—Attributes required for transmitting the BUM traffic. There are two type of tunnels—point-to-multipoint LSP and ingress replication. In the case of ingress replication, the multicast label for BUM traffic is downstream assigned. In the case of point-to-multipoint LSP, the PMSI attribute includes the point-to-multipoint LSP identifier. If the multicast tree is shared or aggregated among multiple EVIs, the PE device uses the upstream assigned label to associate or bind to the EVI.
- **RT Extended Community**—Route target associated with an EVI. This attribute is of global significance in EVPN.

In Figure 97 on page 961, each PE device sends the inclusive multicast route to each BGP neighbor. Device PE1 sends an inclusive multicast route to Devices PE2, PE3, and PE4 for VPN autodiscovery. The handling of BUM traffic is also illustrated in the figure. During the startup sequence, Devices PE1, PE2, PE3, and PE4 send inclusive multicast route that include the multicast label.

Figure 97: VPN Autodiscovery



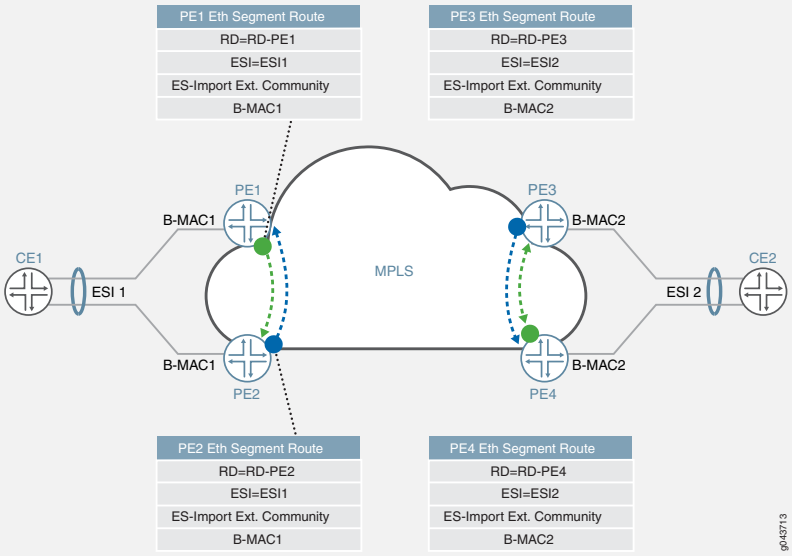
Ethernet Segment Discovery

The Ethernet segment route is encoded in the EVPN NLRI using the Route Type of value 4. This NLRI is used for discovering the multihomed Ethernet segment and for DF election.

ES-import route target, a transitive route target extended community, is also carried with the Ethernet segment route. ES-import extended community enables all the PE devices connected to the same multihomed site to import the Ethernet segment routes. The import of this route is done by the PE device that has the ESI configured. All other PE devices discard this Ethernet segment route.

Figure 98 on page 962 provide details about the procedure of Ethernet segment route for multihomed Ethernet segment autodiscovery.

Figure 98: Ethernet Segment Discovery



In this figure, Devices PE1 and PE2 are connected to a multihomed segment with ESI value of ESI1 and B-MAC address of B-MAC1. In the case of an active-active multihomed segment, this B-MAC should be the on Devices PE1 and PE2. Similarly, Devices PE3 and PE4 are active/active multihomed for ESI2 with B-MAC address of B-MAC2. Devices PE1 and PE2 send the Ethernet segment route for ESI1, which is received by Devices PE3 and PE4, but is ignored because the devices are not configured for ESI1. Only Devices PE1 and PE2 are in one redundant group and the DF election is performed in this group. Similarly, Devices PE3 and PE4 are in another redundant group and either Device PE3 or PE4 is selected as the DF.

Ethernet MAC Routes Discovery

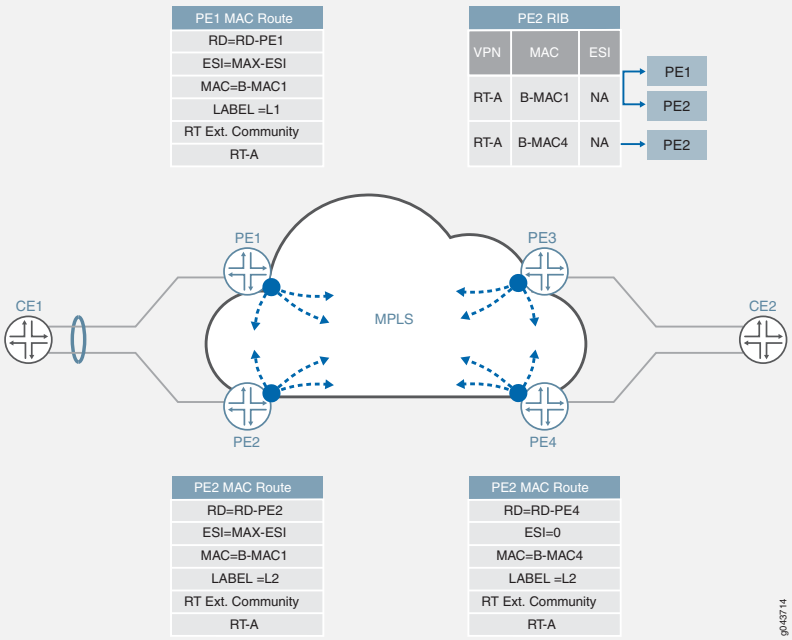
Ethernet MAC Advertisement route is used for distributing B-MAC addresses of PE nodes. The MAC advertisement route is encoded with following fields:

- MAC address field contains B-MAC address.
- Ethernet Tag field is set to 0.
- Ethernet segment identifier field must be set to 0 (for single-homed segments or multihomed segments with per-I-SID load balancing) or to MAX-ESI (for multihomed segment with per-flow load balancing).

- Label is associated with unicast forwarding of traffic from different PE devices.
- RT (route target) extended community associated with its EVI.

Figure 99 on page 963 illustrates the MAC route advertisement in PBB-EVPN.

Figure 99: Ethernet MAC Routes Discovery



Differences Between PBB-EVPN and EVPN

Table 24 on page 963 and Table 25 on page 964 list the differences between PBB-EVPN and pure EVPN for Layer 2 networks in the context of different route types and route attributes, respectively.

Table 24: Route Differences Between PBB-EVPN and EVPN

Route	Usage	Applicability
Ethernet autodiscovery route	• MAC mass withdrawal • Aliasing • Advertising split horizon labels	EVPN only
MAC advertisement route	• Advertise MAC address reachability • Advertise IP and MAC bindings	EVPN PBB-EVPN
Inclusive multicast route	Multicast tunnel endpoint discovery	EVPN PBB-EVPN

Table 24: Route Differences Between PBB-EVPN and EVPN (continued)

Route	Usage	Applicability
Ethernet segment route	• Redundancy • Designated forwarder (DF) election	EVPN PBB-EVPN

Table 25: Route Attributes and Route Usage Differences Between PBB-EVPN and EVPN

Attribute	Usage	Applicability
ESI MPLS lable extended community	• Encode split horizon label for the Ethernet segment. • Indicate redundancy mode (actvie/standby or active/active).	Ethernet autodiscovery route
ES-import extended community	Limit the import scope of the Ethernet segment routes.	Ethernet segment route
MAC mobility extended community	• EVPN: Indicates that a MAC address has moved from one segment to another across PE devices. • PBB-EVPN: Signal C-MAC address flush notification.	MAC advertisement route
Default gateway extended community	Indicate the MAC or IP bindings of a gateway.	MAC advertisement route

PBB-EVPN Packet Walkthrough

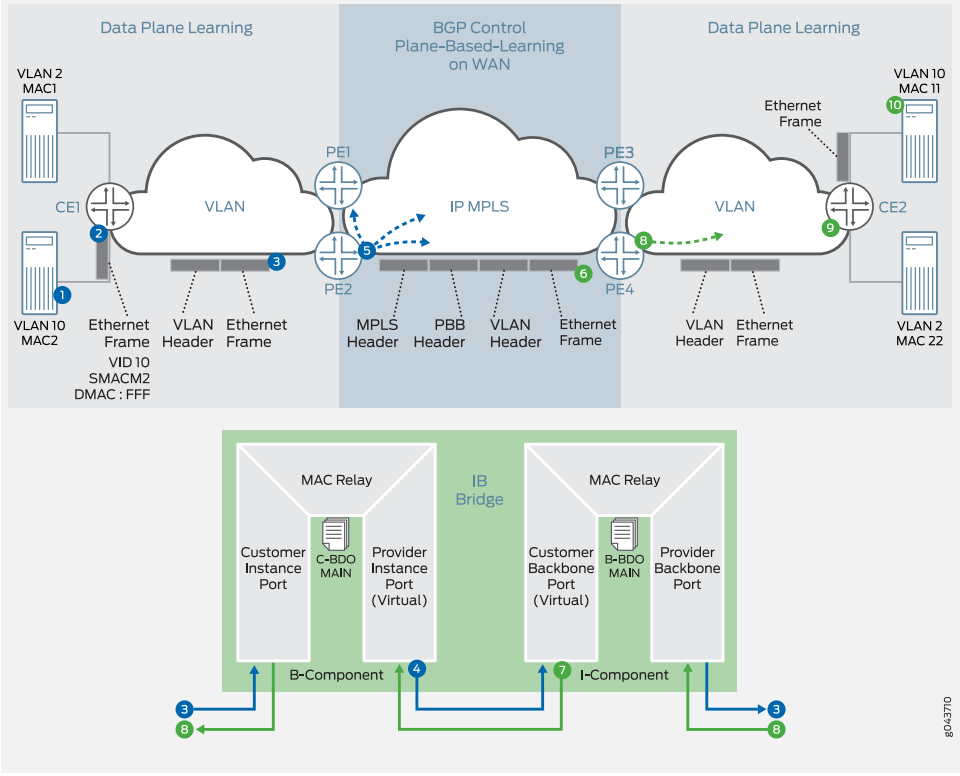
Based on the PBB and EVPN configuration on different PE devices of the network, the Ethernet segment, B-MAC address reachability, and multicast routes are already programmed on different PE devices in the EVPN cloud. The packet walkthrough of PBB-EVPN includes the handling of the following traffic types:

- Handling BUM Traffic in PBB-EVPN on page 964
- Handling Unicast Traffic in PBB-EVPN on page 967
- Handling Path Forwarding in PBB-EVPN on page 969
- Handling MAC Mobility in PBB-EVPN on page 970
- Handling End-to-End OAM for PBB-EVPN on page 971
- Handling QoS and Firewall Policer Support for PBB-EVPN on page 971

Handling BUM Traffic in PBB-EVPN

Figure 100 on page 965 illustrates the PBB-EVPN control plane and BUM traffic handling.

Figure 100: PBB-EVPN BUM Traffic Handling



The PBB-EVPN handling of BUM traffic over the EVPN cloud is as follows:

1. After Server A boots up, Server A attempts to send traffic to Server B. Server A does not have the ARP binding in the ARP table for Server B, so Server A builds an ARP broadcast request and sends it. The contents of the ARP packets are VLAN 10, S-MAC=M2 (server A interface MAC), Destination MAC=ff.ff.ff.ff.ff.ff, source IP address=IP address of Server A or VM IP address, and destination IP address=IP address of Server B. Ether type of packet for ARP is 0x0806. Layer 2 frame is sent to Devie CE1.

2. Device CE1 does the Layer 2 switching operation on this frame. Because this is a Layer 2 broadcast frame, the frame is classified to an interface and based on the bridge domain configuration for this service and broadcast behavior. The packet is forwarded to all the members of the bridge domain except to the one on which it was received. There might be VLAN translation, such as push, pop, or translate, done on this frame. This frame is sent over to Device PE2. This frame might be untagged, single tagged, or Q-in-Q.

3. After Device PE2 receives this frame, at first it goes through the classification engine to classify the frame into a service. Based on the classification result interface (that is, the Customer Instance Port [CIP]) the service is identified. The source MAC address is learned (if it is not present in the MAC table). This classification results in the C-BD.

Because this frame is a broadcast frame, it is sent to all the member interfaces of this bridge domain. One of the member interfaces of this bridge domain is the Provider Instance Port (PIP) interface. Now the packet is formed based on the I-SID configured for this PIP interface. The outer header of the packet at the PIP egress interface is formed based on the following information:

- I-SID—Configured I-SID value on this PIP interface.
- Source MAC address—B-MAC address configured or autogenerated for this frame.
- Destination MAC address—Based on the per-I-SID mapping table that is built based on the source C-MAC-to-B-MAC address learning and the destination C-MAC-to-B-MAC address. For BUM traffic, the default value of the bridge destination address (B-DA) is the Backbone Service Instance Group address. When the B-DA of a frame is a Backbone Service Instance Group address, the normal behavior is to deliver the frame to all Customer Backbone Ports (CBPs) reachable within the backbone VLAN (B-VLAN) to which the backbone service instance is mapped. Filtering based on I-SID by the egress CBP ensures that frames are not transmitted by CBPs that are not part of the backbone service instance.
- Layer 2 Ethernet type—0x88E7.
- Payload—Customer frame.

4. The I-SID-formed packet is sent to the CBP for identifying the backbone bridge domain (B-BD) associated with the I-SID.

5. Lookup in the B-BD is done to send the packet to the right destination. Because this frame is a broadcast frame and the destination B-MAC is a multicast address (00-1E-83-<ISID value>), the packet needs to be handled as the ingress replication (that is, VPLS edge flood next hop) for EVPN. This next hop pushes the service label (multicast MPLS label associated with B-VLAN per peer ID and bridge VLAN ID). The MPLS packet is formed and sent over the MPLS cloud for Devices PE1, PE3 and PE4.

6. The frame is received by Device PE4 as a MPLS packet. The bridge domain identification is accomplished by doing an MPLS label L1 lookup in the mpls.0 routing table. The MPLS lookup points to the table next hop for the bridge domain next hop. After the bridge domain is identified, the packet is identified as a broadcast packet. The BUM composite flood next hop is executed, and this next hop points to the CBP.

7. The egress interfaces are identified. One of the egress interfaces is a PIP interface where the I-SID is configured, and I-SID based filtering (MAC filtering) is applied for filtering the frame. The source C-MAC-to-B-MAC address is learned for the I-SID MAC mapping table. This table is used for building the destination B-MAC address for unicast traffic. The outer I-SID header is popped from the customer Layer 2 frame. The customer bridge domain (C-BD) is found based on the I-SID classification to the PIP interface.

8. The source C-MAC address is learned. The destination C-MAC lookup is done. This is a broadcast frame, and based on the BUM handling (flood next hop), the frame is

forwarded to all the member of the C-BD, except the member interface on which this frame was received.

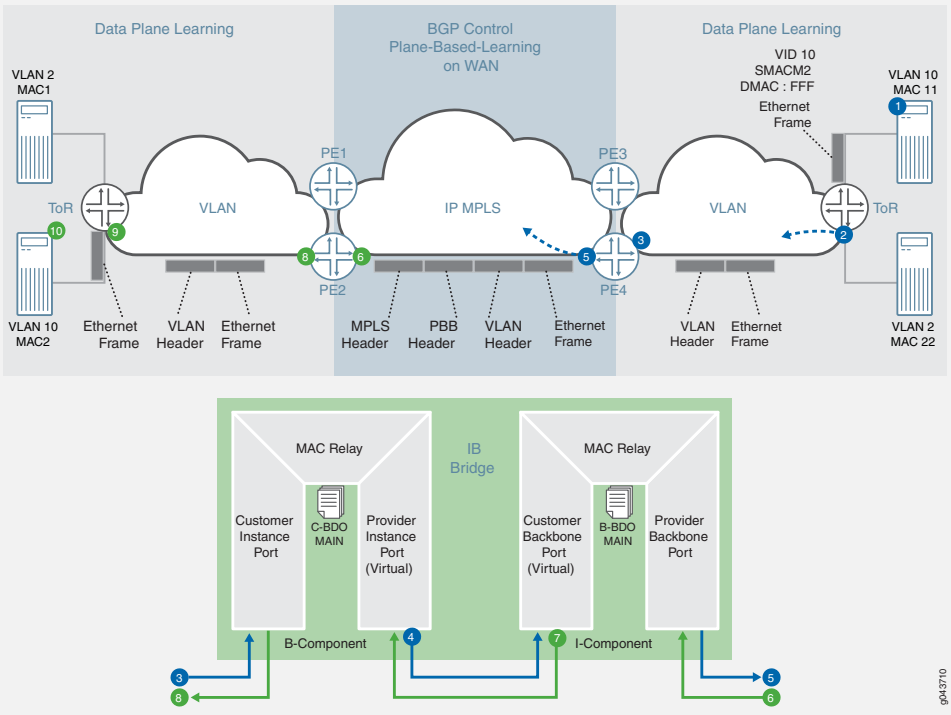
9. Device CE2 receives this frame. Service classification is done based on the frame VLAN. Based on the classification, the bridge domain forwarding service is found and MAC learning is done. Because the frame is a broadcast frame, it is handled by flood next hop.

10. Server B receives the ARP request packet and sends the ARP reply to Server A.

Handling Unicast Traffic in PBB-EVPN

Figure 101 on page 967 illustrates the PBB-EVPN control plane and unicast traffic handling in the form of an ARP reply from Server B.

Figure 101: PBB-EVPN Unicast Traffic Handling



For the unicast traffic flow, it is assumed that both the data and control plane MAC learning has already happened.

1. Server B generates an ARP reply. The contents of the ARP packets are VLAN 10, S-MAC=MAC11 (Server B interface MAC), destination MAC=MACA, source IP address=IP address of Server B or VM IP address, and destination IP address=IP address of Server A. This frame is forwarded to top-of-rack B.

2. After receiving the frame, the CE device classifies the incoming frame. Based on the interface family, the bridge domain associated with the interface is identified. Source MAC address learning takes place on the bridge domain. Next the bridge domain destination MAC (MACA) lookup is done, and the lookup provides the Layer 2 egress interface. The output feature of the egress interface is applied before the CE device sends the frame to the egress interface.

3. Layer 2 encapsulated frame is received by Device PE4. The Layer 2 service classification is done to identify the customer bridge domain (C-BD) associated with this frame. The source MAC address (MAC11) learning is done in the context of the C-BD on the CIP interface.

4. Destination MAC lookup in the context of C-BD points to the PIP interface. At this point , the PIP interface egress feature list is executed. Based on the feature list, the outer I-SID header is pushed on the original Ethernet frame.
   - Source MAC—B-MAC of Device PE4
   - Destination MAC—B-MAC of Device PE2 (lookup result in I-SID C-MAC-to-B-MAC table)
   - I-SID—Configured value of I-SID
   - Layer 2 Ether typ—0x88E7
5. The destination MAC address (B-MAC of Device PE2) lookup is done in the B-BD MAC address table. This lookup results in a unicast next hop (that is, an EVPN next hop). This next hop contains a unicast MPLS service label. This label is distributed through the multiprotocol BGP (MP-BGP) control plane. The downstream peer allocates this MPLS service label. Allocation of this label can be per EVI; per EVI and VLAN; per EVI, VLAN, and attachment circuit; or per MAC address. Based on the information in the next hop, the MPLS packet is formed and forwarded on the MPLS network.

6. Device PE2 receives the frame. It is identified as an MPLS packet. The MPLS label lookup is done in the mpls.0 routing table. This lookup results in the table next hop. This lookup results in the B-BD table. The B-MAC rule (that is, source B-MAC is destination B-MAC) and I-SID filtering (CBP configured ISID=packet ISID) rules are applied. Based on the received frame I-SID, CBP is identified and B-VLAN is popped.

7. The frame header is passed to the PIP interface for further processing. The C-MAC address (M11 to B-MAC-PE2) to B-MAC mapping is learned in the I-SID table. The outer I-SID header is popped.

8. The inner source MAC address is learned on the PIP interface in the context of C-BD. The inner destination MAC address lookup is done, resulting in the egress CIP interface.

9. The CE device receives the Layer 2 frame, and Layer 2 forwarding is done.

10. Server A receives the unicast ARP reply packet from Server B.

Handling Path Forwarding in PBB-EVPN

In a PBB-EVPN network, a frame can come from either the customer edge (CE) side (bridge interface) or the MPLS-enabled interface (core-facing interface).

The packet flow for packets received from the CE side is as follows:

1. If the frame is received from a CE interface, the interface belongs to the bridge family, and the MAC address lookup and learning are done in the customer bridge domain (C-BD) context. The result of the lookup is a unicast MAC route or a flood MAC route.

2. The next lookup is done in the I-SID MAC table to determine the destination B-MAC associated with the destination C-MAC.

3. The I-SID header is prepended to the packet.

4. The next lookup is done in the B-BD because the PIP interface belongs to the bridge family.

5. The B-BD lookup points to either the unicast MAC route or the flood MAC route and this route points to either the EVPN indirect multicast next hop or the unicast indirect next hop.

The packet flow for packets received from the core side is as follows:

1. If the frame is received from the core-facing interface, the interface belongs to the MPLS family, and the MPLS label lookup is done in the mpls.0 routing table next hop. The result of this lookup is the routing instance context.

2. The next lookup is done based on the I-SID from the packet to the BBD lookup.

3. If the BBD is found, then I-SID based filtering rules are applied, where the I-SID configured MAC should match the packet source B-MAC, then the frame is dropped.

4. The I-SID MAC table is updated for the destination B-MAC associated with the destination C-MAC for building the C-MAC-to-B-MAC association.

5. The I-SID header is removed and C-BD is found based on the PIP interface.

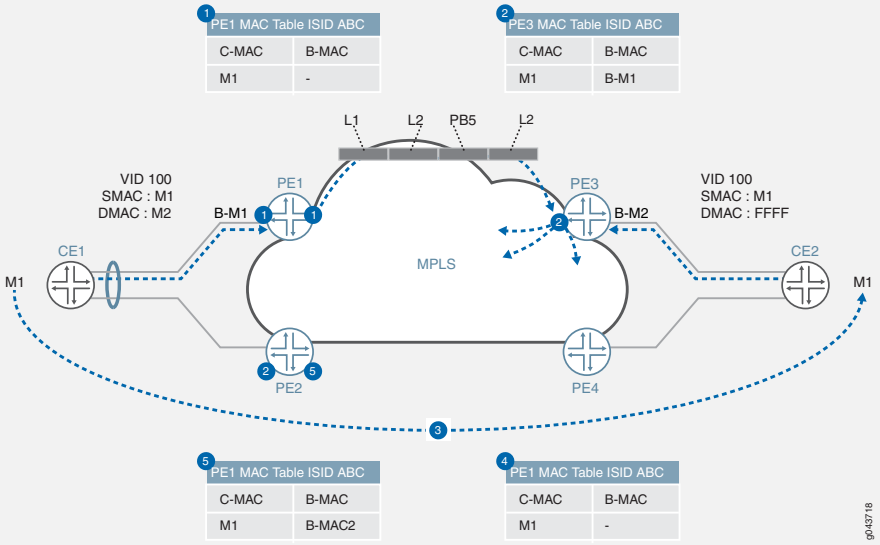
6. The next lookup is done in the C-BD because the PIP interface belongs to the bridge family.

7. The C-BD lookup points to either a unicast MAC route or a flood MAC route, and this route points to a CE interface or flood route.

Handling MAC Mobility in PBB-EVPN

Figure 102 on page 970 illustrates PBB-EVPN MAC mobility from the point of the forwarding and control plane.

Figure 102: PBB-EVPN MAC Mobility Handling



MAC mobility from the point of the forwarding and control plane is handled as follows:

1. Device PE1 learns the C-MAC address, M1, on the local port and forwards it across the core according to the destination-C-MAC-to-remote-B-MAC mapping. This mapping is either statically configured or learned through the data plane. If the destination-C-MAC-to-remote-B-MAC mapping is not found in the I-SID mapping table, then the remote B-MAC is derived by using the I-SID.

2. Device PE3 learns the C-MAC address, M1, through the B-MAC address, B-M1, from the dataplane.

3. Customer M1 is moved from Device CE1 to behind Device CE2.

4. When Customer M1 wants to communicate with a customer behind Device CE1, a broadcast traffic with VID: 100, Source MAC: M1, and destination MAC: ff.ff.ff.ff.ff.ff is

sent. Device PE3 learns the MAC M1 in the C-BD MAC table and updates the M1 location in the I-SID mapping table.

5. Device PE1 receives the packet and M1 is learned and updated in the I-SID mapping table as reachable through the remote MAC is B-M2.

Handling End-to-End OAM for PBB-EVPN

You can run provider-level Operation, Administration, and Maintenance (OAM) by running connectivity fault management (CFM) on the inward or outward facing maintenance endpoints (MEPs) either on the PIP interface or over the EVPN service.

Currently, the designated forwarder (DF) election is decided based on the DF election algorithm and it is the local decision of the PE devices. This is useful in an end-to-end service handling scenario, where the decision of DF election can be done with operator's consent as well and vice versa. Another scenario in which it might be useful to influence the DF role per service basis or propagating the DF to a CE device is for multihomed networks, where there is no direct link between the CE and PE devices.

Handling QoS and Firewall Policer Support for PBB-EVPN

Table 26 on page 971 provides details about the QoS and firewall features that are supported in the context of PBB-EVPN integration.

Table 26: Firewall and QoS Feature Support in PBB-EVPN

Feature	Description	Support on round-trip time (RTT)	Support on CE Interface	Support on Core Interface
Classification	Fixed classification to one FC	Yes	Yes	Yes
	Behavior aggregate (BA) and multifield classifier (MF) classification for inner output vlan .1p bit	Yes	Yes	Yes
	BA and MF classification based on DEI and PCP	No	Not required	No
	BA and MF classification based on Exp	No	No	Yes
CoS Marking	.1P to I-SID PCP and DEI: Customer VLAN .1p	No	By default, .1P is mapped to PCP and DEI.	Yes
	.1P to Exp: Customer VLAN .1p	No	No	Yes
	MPLS EXP to I-SID PCP and DEI	No	Default behavior	No
	EXP to .1P	No	Yes	No

Table 26: Firewall and QoS Feature Support in PBB-EVPN (continued)

Feature	Description	Support on round-trip time (RTT)	Support on CE Interface	Support on Core Interface
QoS shaping	Hierarchical scheduling and shaping on ingress device	No	Yes	Yes
	Hierarchical scheduling and shaping on egress device	No	Yes	Yes
Firewall filter	Filtering BUM traffic	Unknown traffic only	Broadcast and multicast traffic only	Broadcast and multicast traffic only
	I-SID-based firewall filter	No	No	No
	Customer VLAN-based filter	No	Yes	Yes
Policer (2 rate 3 color)	Ingress direction	No	Yes	Yes
	Egress direction	No	Yes	Yes

Implementation Overview of PBB-EVPN Integration

The following sections provide use case scenarios for PBB-EVPN integration for DCI.

- PBB-EVPN Failure Scenarios on page 972
- PBB-EVPN I-SID Use Case Scenarios on page 975
- VPLS Integration with PBB-EVPN Use Case Scenario on page 976
- PBB-EVPN Redundancy Use Case Scenarios on page 977

PBB-EVPN Failure Scenarios

There are different PBB-EVPN failure scenarios that should be taken care of while providing an end-to-end solution. These failure scenarios can be of the following types:

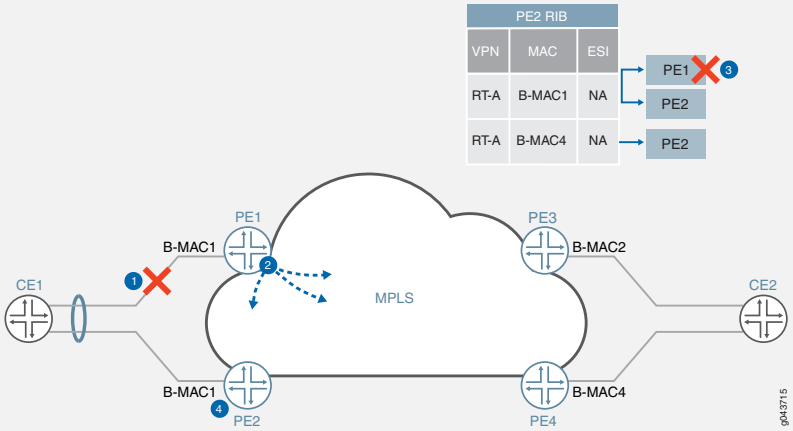
- Segment Failure on page 972
- Node Failure on page 973
- Core Failure on page 974

Segment Failure

A segment, or CE-facing link, failure is handled in the activce/active and active/standby multihoming redundancy modes.

Figure 103 on page 973 shows the handling of segment failure for flow-based load balancing at Device CE1.

Figure 103: PBB-EVPN Segment Failure



A segment failure in PBB-EVPN is handled as follows:

1. Ethernet link between Devices CE1 and PE1 failed due to fiber cut or the interface is down. Device PE1 detects the failed segment.

2. Device PE1 withdraws the B-MAC address that is advertised for the failed segment (B-M1).

3. The CE1-facing link goes down. If the link failure happens in the single-active redundancy mode or no redundancy case, the C-MAC flush is also done.

   The C-MAC address flushing happens in two ways:

   - If Device PE2 uses the shared B-MAC address for multiple I-SIDs, it notifies the remote PE device by readvertising the B-MAC address with the MAC mobility extended community attribute by incrementing the value of counter. This causes the remote PE device to flush all C-MAC addresses associated with the B-MAC address for Device PE1.
   - If Device PE2 uses the dedicated B-MAC address, then it withdraws the B-MAC address associated with the failed segment and sends it to Devices PE2, PE3, and PE4.

4. After receiving the B-MAC withdraw from Device PE1, Device PE3 removes PE1 reachability for B-MAC1 from its forwarding table. Reachability of B-MAC1 through Device PE2 still exists.

5. The DF election is rerun on Device PE2 for all the I-SIDs for the Ethernet segment ESI.

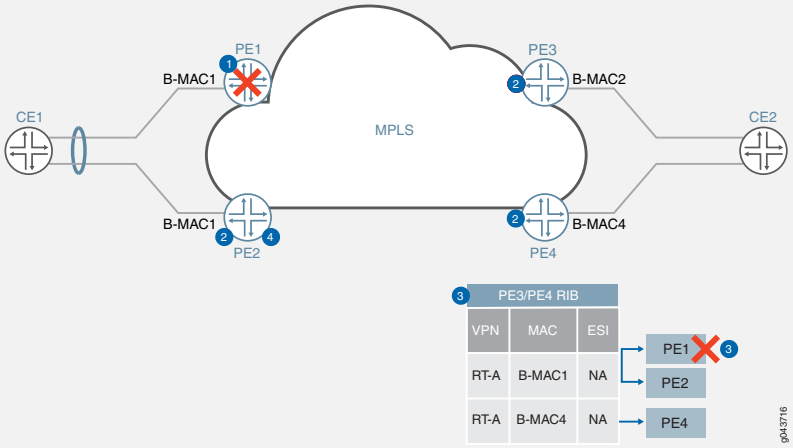
Node Failure

A node, or PE device, failure scenario is similar to a segment failure from the point of view of CE side failure handling, but it is different from core side failure handling. In the case

of core side failure handling, EVPN depends on the BGP session timeout for clearing the state of the EVPN sessions on affected PE devices.

Figure 104 on page 974 illustrates a node failure scenario for node failure handling.

Figure 104: PBB-EVPN Node Failure



1. Device PE1 failed and the CE side switchover to Device PE2 is done by an interface down event.

2. Devices PE2, PE3, and PE4, or BGP route reflector, detect the BGP session timeout with Device PE1.

3. As soon as the BGP session timeout happens, Devices PE3 and PE4 remove Device PE1 from the forwarding table by marking the Device PE1 next hop as unreachable or deleted. In the case of single-active redundancy mode, the I-SID table for C-MAC-to-B-MAC mapping table has to be flushed or updated. In the case of active/active redundancy mode, it is not required to flush the I-SID table, because the same B-MAC address is used for both Devices PE1 and PE2 for a given EVI.

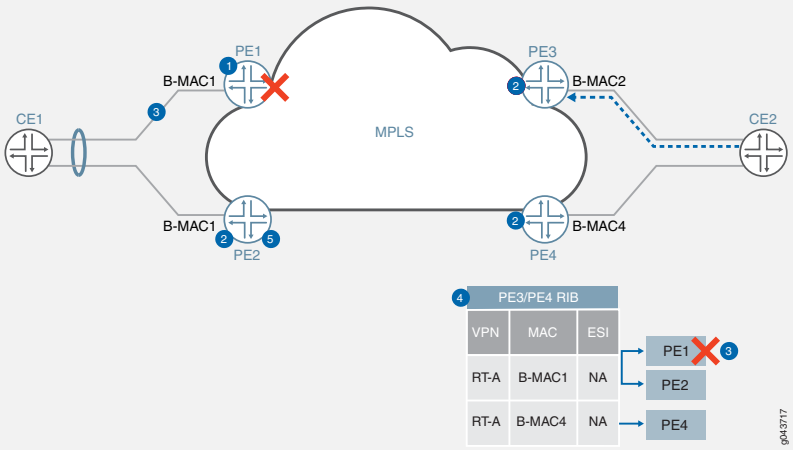
4. At Device PE2, after a BGP timeout, the DF election algorithm is rerun and Device PE2 becomes the DF for all I-SIDs on an affected Ethernet segment.

Core Failure

The handling of core side isolation in the EVPN network is similar to the PE side failure, with some differences in handling of the CE device or Ethernet segment.

Figure 105 on page 975 provides details about the handling of core isolation.

Figure 105: PBB-EVPN Core Failure



Core isolation in PBB-EVPN is handled as follows:

1. Device PE1 looses connectivity to the core.

2. Devices PE2, PE3, and PE4, or BGP route reflector, detect the BGP session timeout with Device PE1.

3. Device PE1 sends an LACP OUT_OF_SYNC message to Device CE1 to take the port out of the bundle.

4. Device PE2, or BGP route reflector, detects the BGP session timeout with Device PE1.

5. Device PE2 reruns the DF election and is selected as the DF for all the I-SIDs on the segment.

PBB-EVPN I-SID Use Case Scenarios

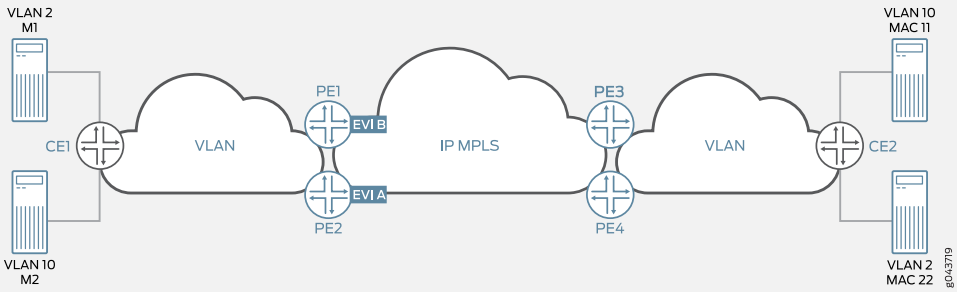
- I-SID Base Service on page 975
- I-SID-Aware Service on page 976

I-SID Base Service

In the case of the I-SID base service, there is one-to-one mapping between a bridge domain and an EVI. In this case, there is no need to carry the I-SID in the MAC advertisement route, because the bridge domain ID can be derived from the route target (RT) associated with this route. The MPLS label allocation is done on a per-EVI basis.

Figure 106 on page 976 provides an overview of the I-SID base use case scenario.

Figure 106: I-SID Base Service Use Case



In the case of I-SID load balancing, where load balancing of traffic is done on a per-service basis from the originating CE link aggregation group (LAG) configuration, there are two models for B-MAC addresses:

- Shared source B-MAC

  In this model, all I-SIDs from an Ethernet segment share one source B-MAC address. This model has limitations from the point of view of B-MAC withdrawal because of service failure. The remote PE device needs to flush B-MAC-to-C-MAC mapping for all I-SIDs . This creates problem for convergence because MAC flush is done for all I-SIDs.

- Unique source B-MAC per I-SID

  Unique unicast B-MAC addresses (one per I-SID) are allocated per multihomed Ethernet segment. The DF filtering is applied to unicast and multicast traffic, in both the core-to-segment and segment-to-core directions.

I-SID-Aware Service

In the case of I-SID-aware service, multiple I-SIDs can be mapped to the same EVI. But there is one-to-one mapping between a bridge domain and an I-SID. The Ethernet Tag ID must be set to the I-SID in the BGP routes advertisements. The MPLS label allocation is done on a per-EVI or per-EVI/I-SID basis so that the PBB can be terminated at the ingress PE device and recreated at the egress PE device.

VPLS Integration with PBB-EVPN Use Case Scenario

In this use case scenario, VPLS is one cloud that is getting integrated with PBB-EVPN by using logical tunnel interfaces. The logical tunnel interface is being terminated into the customer bridge domain (C-BD). MAC address learning from the VPLS cloud is happening in the context of the C-BD. The C-bridge domain gets mapped to the backbone bridge domain and going to the EVPN cloud.

NOTE: PBB-VPLS interworking with PBB-EVPN under an EVI is not supported in Junos OS Release 17.2R1.

PBB-EVPN Redundancy Use Case Scenarios

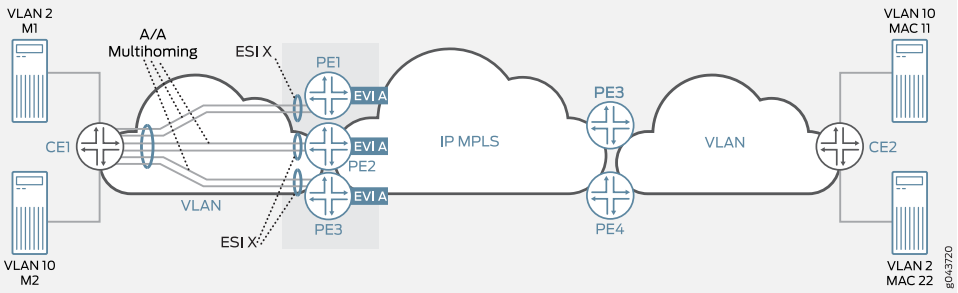
Single Active Redundancy Use Case Scenario

In this use case scenario, a CE device is multihomed to multiple PE devices. The Ethernet segment for this scenario is defined by configuring the same ESI ID on multiple physical interfaces or aggregated Ethernet interfaces on the PE devices along with the mode of operation. In this mode of operation, only one PE device (that is, the DF) is allowed to forward the traffic to and from this Ethernet segment for BUM traffic. The DF election is done based on each ESI in an EVI and by taking the lowest configured I-SID into consideration. This also depends on this number of PE devices to which a segment is multihomed. The service carving is achieved by putting different I-SIDs in different EVIs. The DF election is similar to the DF election in the VLAN-based EVPN. A default timer of 3 seconds is used for reception of Ethernet segment routes from other PE nodes, and this timer can be configured with the same command as used in EVPN—the **designated-forwarder-election hold-time** statement.

In case of PBB-EVPN, the Ethernet autodiscovery route is not used. This mode is specified in the B-MAC advertisement route. In a single-homed or multihomed setup, with the single-active mode, the ESI field must be set to 0 in the B-MAC route advertisement. If ESI 0 is used in the MAC advertisement route, then I-SID-based load balancing is performed. An I-SID value can serve as a single home, an active/standby scenario, or an active/active scenario. It cannot, however, be used in a mixed mode of operation.

Figure 107 on page 977 provides the use case scenario for active/standby multihoming along with DF election.

Figure 107: PBB-EVPN Redundancy Use Case



Active/Standby Redundancy Use Case Scenario

In the case of the active/active redundancy use case scenario, the DF election is used for handling the BUM traffic. In the case of PBB-EVPN, split horizon of EVPN is not used for filtering the BUM traffic. Instead, BUM traffic is filtered by filtering the destination B-MAC, where the configured B-MAC is the same as the received packet B-MAC; therefore that packet is from the same segment.

The aliasing approach is the same as the EVPN, but the B-MAC route is advertised by setting the ESI field to MAX-ESI. When the remote PE device receives the B-MAC route with the MAX-ESI value, then the remote PE device does the load balancing between Devices PE1 and PE2.

Active/Active Redundancy Use Case Scenario

In a PBB-EVPN active-active multihomed network, all the multihomed PE devices require identical MAC installations. For this purpose, BGP is used to sync the source C-MAC addresses (on the CE side) or the remote C-MAC addresses (from the core) across the multihomed PE devices for same ESI.

To enable MAC sync:

1. For the source C-MAC address sync:
   - Configure per-packet-load-balancing on the CE device.
   - Ensure that there are minimum flows per source C-MAC, so that each source C-MAC takes both links toward the multihomed PE devices at least once. This ensures that both the multihomed PE devices learn each source C-MAC.
2. For remote C-MAC address sync:
   - Ensure that there are minimum flows per remote C-MAC, so that each remote C-MAC takes both links(aliasing) towards the multihomed PE devices at least once while traversing the core. This ensures that both the multihomed PE devices learn each remote C-MAC.

Configuration Overview of PBB-EVPN Integration

The PBB-EVPN configuration is done using the following models:

- One-to-one mapping between the I-SID and the bridge domain

  In this configuration model, there is a shared EVPN instance (EVI) between different services, although there is one-to-one mapping between the bridge domain and the I-SID.

- Many-to-one mapping between the I-SID and the bridge domain

  In this configuration model, virtual switch configuration is used to allow multiple I-SIDs to be mapped to one bridge domain. This model allows only one bridge domain in a given EVI, and all other bridge domains are mapped to the other Layer 2 services.

Sample PBB-EVPN Port Configuration:

- Provider Backbone Port (PBP) Configuration:

```
routing-instances {
   evpnA {
      instance-type virtual-switch;
      route-distinguisher 192.0.2.1;
      vrf-target target:65221:111;
      protocols {
```

```
        evpn {
          label-allocation per-instance;
          extended-isid-list [10000 10401 20000-20001]
        }
      }
    }
  }
```

- Customer Backbone Port (CBP) Configuration:

```
interfaces {
  cbp {
    flexible-valn-tagging;
    unit 0 {
      family bridge {
        interface-mode trunk;
        isid-list [10000 10401 20000-20001] [all];
      }
    }
  }
}
```

- Provider Instance Port (PIP) Configuration:

```
interfaces {
  pip {
    flexible-valn-tagging;
    unit 0 {
      family bridge {
        interface-mode trunk;
        isid-list [20000 20001];
      }
    }
    unit 1 {
      family bridge {
        interface-mode trunk;
        isid-list [10000 10401];
      }
    }
  }
}
```

- Customer Instance Port (CIP) Configuration:

```
interfaces {
  ge-4/0/0 {
    flexible-valn-tagging;
  esi 1 primary;
    unit 0 {
      family bridge {
        interface-mode trunk;
        vlan-id-list [1-399 500 800];
      }
    }
```

```
    ge-7/0/0/ {
      flexible-vlan-tagging;
      unit 0 {
        family bridge {
          interface-mode trunk;
          vlan-id-list [1-401];
        }
      }
    }
    ge-8/0/0/ {
      flexible-vlan-tagging;
      unit 0 {
        family bridge {
          interface-mode trunk;
          vlan-id-list [500-501];
        }
      }
    }
}
```

Sample PBB-EVPN Routing Instance Configuration:

- Provider Routing Instance Configuration:

```
routing-instances {
  EVPN A {
    instance-type virtual-switch;
    route-distinguisher 192.0.2.1;
    vrf-target target:65221:111;
    protocols {
      pbb-evpn {
        label-allocation per-instance;
        extended-isid-list [10000 10401 20000] [all];
      }
      evpn {
        label-allocation per-instance;
        extended-vlan-list 1000;
      }
    }
    interface cbp.0;
    bridge-domains {
      B-BD1 {
        isid-list 10000;
      }
      B-BD2 {
        isid-list 10401;
      }
      B-BD3 {
        isid-list 20000;
      }
      B-BD4 {
        isid-list 1000;
      }
    }
```

```
    pbb-options {
      vlan-id 1000 isid-list [20001];
      default-bvlan 22;
    }
  }
}
```

- Customer Routing Instance Configuration:

```
routing-instances {
  PBN-1 {
    instance-type virtual-switch;
    interface ge-4/0/0.0;
    interface pip.0;
    bridge-domains {
      customer-BDs {
        vlan-id-list [1-399 500 800];
      }
    }
    pbb-options {
      peer-instance EVPN A;
      source-bmac <mac-address>;
      service-groups {
        pbb-1-isid {
          source-bmac <mac-address>;
          isid {
            isid-1 {
              isid 20000 vlan-id-list [1-399 500] [all];
              source-bmac <mac-address>;
              map-dest-bmac-to-dest-cmac <b-mac> <c-mac>;
              }
              isid-2 {
                isid 20001 vlan-id-list [800] [all];
              }
            }
          }
        }
    }
  }
  PBN-2 {
    instance-type virtual-switch;
    interface ge-7/0/0.0;
    interface pip.1;
    bridge-domains {
      customer-BDs {
        vlan-id-list [1-401];
      }
    }
    pbb-options {
      peer-instance EVPN A;
      default-isid <i-sid>;
      service-groups {
        pbb-2-isid-1 {
          isid {
            isid-1 {
```

```
                isid 10000 vlan-id-list [1-400];
                }
              }
            }
          }
          pbb-2-isid-2 {
            isid {
              isid-1 {
                isid 10401 vlan-id-list [401];
              }
            }
          }
      }
    }
}
```

Supported and Unsupported Features on PBB-EVPN

Junos OS supports the following features with PBB-EVPN:

- Graceful Routing Engine switchover (GRES), unified in-service software upgrade (ISSU), and nonstop software upgrade (NSSU).
- Nonstop active routing (NSR) for BGP peers configured with EVPN family.

  NSR on PBB-EVPN replicates and recreates backbone MAC (B-MAC) routes, inclusive multicast routes, and Ethernet segment identifier (ESI) routes.
- Feature support on 64-bit platforms.
- IEEE assigned standard ether type as 0x88E7 for I-SID frames. In addition to this, 802.1x can be used.

  The following security considerations are supported:

  - Packet destined to the Layer 2 ether type as 0x88E7 is processed only if PBB is enabled on the ingress core PE device.
  - Packet received from the core is processed only if the I-SID is known and configured on the ingress PE device; otherwise, the frame is dropped.

Junos OS does not support the following features for PBB-EVPN integration:

- Complete support of EVPN NSR
- Integrated routing and bridging (IRB) interfaces
- IPv6 IP addresses
- Logical systems

Related Documentation

- Example: Configuring PBB with Multihomed EVPN on page 1005
- Example: Configuring PBB with Single-Homed EVPN on page 983
- pbb-evpn-core on page 1082

Example: Configuring PBB with Single-Homed EVPN

This example shows how to integrate provider backbone bridging (PBB) with Ethernet VPN (EVPN). With this integration, the control plane operations in the core are simplified, providing faster convergence and scalability enhancements than regular EVPN. The PBB-EVPN applications include Data Center Interconnect (DCI) and carrier Ethernet E-LAN services.

Requirements

This example uses the following hardware and software components:

- Three provider edge (PE) devices each connected to single-homed customer sites.
- Four customer edge (CE) devices that are single-homed to the PE devices.
- Junos OS Release 17.2R1 or later running on all the PE routers.

Before you begin:

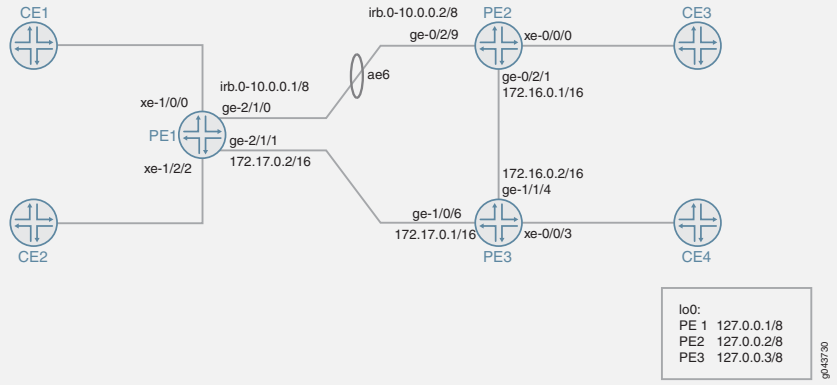
- Configure the device interfaces.
- Configure an IGP, such as OSPF, on all the PE devices.
- Establish an internal BGP session between the PE devices.
- Enable RSVP on the PE devices.
- Configure MPLS and label-switched paths (LSPs) between the PE devices.

Overview and Topology

Starting in Junos OS Release 17.2R1, PBB is integrated with Ethernet VPN (EVPN) to enable significant reduction in the control plane learning across the core, allowing a huge number of Layer 2 services, such as data center connectivity, to transit the network in a simplified manner.

In a PBB-EVPN network, the backbone core bridge (BCB) device in the PBB core is replaced with MPLS, while retaining the service scaling properties of the PBB backbone edge bridge (BEB). The B-component (provider routing instance) is signalled using EVPN BGP signaling and encapsulated inside MPLS using provider edge (PE) and provider (P) devices. Thus, PBB-EVPN combines the vast scaling property of PBB with the simplicity of a traditional basic MPLS core network, resulting in significant reduction in the amount of network-wide state information, as opposed to regular PBB.

Figure 108: PBB with Single-Homed EVPN



In Figure 108 on page 984, PBB is integrated with EVPN, where the CE devices are single-homed to Devices PE1, PE2, and PE3.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

PE1

```
set chassis aggregated-devices ethernet device-count 16
set chassis network-services enhanced-ip
set interfaces xe-1/0/0 flexible-vlan-tagging
set interfaces xe-1/0/0 encapsulation flexible-ethernet-services
set interfaces xe-1/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-1/0/0 unit 0 vlan-id 10
set interfaces xe-1/0/0 unit 1 encapsulation vlan-bridge
set interfaces xe-1/0/0 unit 1 vlan-id 20
set interfaces xe-1/2/2 flexible-vlan-tagging
set interfaces xe-1/2/2 encapsulation flexible-ethernet-services
set interfaces xe-1/2/2 unit 0 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 0 vlan-id 10
set interfaces xe-1/2/2 unit 0 family bridge filter input BRI
set interfaces xe-1/2/2 unit 1 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 1 vlan-id 20
set interfaces xe-1/2/2 unit 2 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 2 vlan-id 11
set interfaces xe-1/2/2 unit 2 family bridge
set interfaces xe-1/2/2 unit 3 encapsulation vlan-bridge
set interfaces xe-1/2/2 unit 3 vlan-id 21
set interfaces xe-1/2/2 unit 3 family bridge
set interfaces ge-2/1/0 gigether-options 802.3ad ae6
set interfaces ge-2/1/1 unit 0 family inet address 10.0.0.1/8
set interfaces ge-2/1/1 unit 0 family iso
set interfaces ge-2/1/1 unit 0 family mpls
set interfaces ae6 encapsulation ethernet-bridge
```

```
set interfaces ae6 unit 0 family bridge
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces irb unit 0 family inet address 10.0.0.1/8
set interfaces irb unit 0 family iso
set interfaces irb unit 0 family mpls
set interfaces lo0 unit 0 family inet address 127.0.0.1/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.1
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE1toPE2 from 127.0.0.1
set protocols mpls label-switched-path PE1toPE2 to 127.0.0.2
set protocols mpls label-switched-path PE1toPE3 from 127.0.0.1
set protocols mpls label-switched-path PE1toPE3 to 127.0.0.3
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.1:100
set routing-instances pbbn1 vrf-target target:100:100
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 protocols evpn extended-isid-list 2000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbbn1 bridge-domains bdb vlan-id 200
set routing-instances pbbn1 bridge-domains bdb isid-list 2000
set routing-instances pbbn1 bridge-domains bdb vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-1/2/2.0
set routing-instances pbn1 bridge-domains bda interface xe-1/0/0.0
set routing-instances pbn1 bridge-domains bdb domain-type bridge
```

```
set routing-instances pbn1 bridge-domains bdb vlan-id 20
set routing-instances pbn1 bridge-domains bdb interface xe-1/2/2.1
set routing-instances pbn1 bridge-domains bdb interface xe-1/0/0.1
set routing-instances pbn1 bridge-domains bdc domain-type bridge
set routing-instances pbn1 bridge-domains bdc vlan-id 11
set routing-instances pbn1 bridge-domains bdc interface xe-1/2/2.2
set routing-instances pbn1 bridge-domains bdd domain-type bridge
set routing-instances pbn1 bridge-domains bdd vlan-id 21
set routing-instances pbn1 bridge-domains bdd interface xe-1/2/2.3
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  11
set routing-instances pbn1 service-groups sga pbb-service-options source-bmac
  00:50:50:50:50:50
set routing-instances pbn1 service-groups sgb service-type elan
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  20
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  21
set bridge-domains bd vlan-id none
set bridge-domains bd interface ae6.0
set bridge-domains bd routing-interface irb.0
```

PE2

```
set chassis aggregated-devices ethernet device-count 3
set chassis network-services enhanced-ip
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation flexible-ethernet-services
set interfaces xe-0/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 0 vlan-id 10
set interfaces xe-0/0/0 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 1 vlan-id 20
set interfaces xe-0/0/0 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 2 vlan-id 11
set interfaces xe-0/0/0 unit 2 family bridge
set interfaces xe-0/0/0 unit 3 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 3 vlan-id 21
set interfaces xe-0/0/0 unit 3 family bridge
set interfaces ge-0/2/1 unit 0 family inet address 172.16.0.1/16
set interfaces ge-0/2/1 unit 0 family mpls
set interfaces ge-0/2/9 gigether-options 802.3ad ae6
set interfaces ae6 encapsulation ethernet-bridge
set interfaces ae6 unit 0 family bridge
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces irb unit 0 family inet address 10.0.0.2/8
set interfaces irb unit 0 family mpls
```

```
set interfaces lo0 unit 0 family inet address 127.0.0.2/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.2
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE2toPE1 from 127.0.0.2
set protocols mpls label-switched-path PE2toPE1 to 127.0.0.1
set protocols mpls label-switched-path PE2toPE3 from 127.0.0.2
set protocols mpls label-switched-path PE2toPE3 to 127.0.0.3
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.2:100
set routing-instances pbbn1 vrf-target target:100:100
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 protocols evpn extended-isid-list 2000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbbn1 bridge-domains bdb vlan-id 200
set routing-instances pbbn1 bridge-domains bdb isid-list 2000
set routing-instances pbbn1 bridge-domains bdb vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-0/0/0.0
set routing-instances pbn1 bridge-domains bdb domain-type bridge
set routing-instances pbn1 bridge-domains bdb vlan-id 20
set routing-instances pbn1 bridge-domains bdb interface xe-0/0/0.1
set routing-instances pbn1 bridge-domains bdc domain-type bridge
set routing-instances pbn1 bridge-domains bdc vlan-id 11
set routing-instances pbn1 bridge-domains bdc interface xe-0/0/0.2
set routing-instances pbn1 bridge-domains bdd domain-type bridge
set routing-instances pbn1 bridge-domains bdd vlan-id 21
set routing-instances pbn1 bridge-domains bdd interface xe-0/0/0.3
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
```

```
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  11
set routing-instances pbn1 service-groups sga pbb-service-options source-bmac
  00:51:51:51:51:51
set routing-instances pbn1 service-groups sgb service-type elan
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  20
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  21
set bridge-domains bd vlan-id none
set bridge-domains bd interface ae6.0
set bridge-domains bd routing-interface irb.0
```

PE3

```
set chassis aggregated-devices ethernet device-count 16
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 1 vlan-id 20
set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 2 vlan-id 11
set interfaces xe-0/0/3 unit 2 family bridge
set interfaces xe-0/0/3 unit 3 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 3 vlan-id 21
set interfaces xe-0/0/3 unit 3 family bridge
set interfaces ge-1/0/6 unit 0 family inet address 172.17.0.1/16
set interfaces ge-1/0/6 unit 0 family mpls
set interfaces ge-1/1/4 unit 0 family inet address 172.16.0.2/16
set interfaces ge-1/1/4 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.3/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.3
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE3toPE1 from 127.0.0.3
set protocols mpls label-switched-path PE3toPE1 to 127.0.0.1
set protocols mpls label-switched-path PE3toPE2 from 127.0.0.3
```

```
set protocols mpls label-switched-path PE3toPE2 to 127.0.0.2
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.3
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.3:100
set routing-instances pbbn1 vrf-target target:100:100
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 protocols evpn extended-isid-list 2000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbbn1 bridge-domains bdb vlan-id 200
set routing-instances pbbn1 bridge-domains bdb isid-list 2000
set routing-instances pbbn1 bridge-domains bdb vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-0/0/3.0
set routing-instances pbn1 bridge-domains bdb domain-type bridge
set routing-instances pbn1 bridge-domains bdb vlan-id 20
set routing-instances pbn1 bridge-domains bdb interface xe-0/0/3.1
set routing-instances pbn1 bridge-domains bdc domain-type bridge
set routing-instances pbn1 bridge-domains bdc vlan-id 11
set routing-instances pbn1 bridge-domains bdc interface xe-0/0/3.2
set routing-instances pbn1 bridge-domains bdd domain-type bridge
set routing-instances pbn1 bridge-domains bdd vlan-id 21
set routing-instances pbn1 bridge-domains bdd interface xe-0/0/3.3
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  11
set routing-instances pbn1 service-groups sga pbb-service-options source-bmac
  00:52:52:52:52:52
set routing-instances pbn1 service-groups sgb service-type elan
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  20
set routing-instances pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  21
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device PE1:

1. Set the number of aggregated Ethernet interfaces on Device PE1.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 16
```

2. Set Device PE1's network services to enhanced Internet Protocol and use enhanced mode capabilities.

```
[edit chassis]
user@PE1# set chassis network-services enhanced-ip
```

3. Configure the CE-facing interfaces of Device PE1.

```
[edit interfaces]
user@PE1# set xe-1/0/0 flexible-vlan-tagging
user@PE1# set xe-1/0/0 encapsulation flexible-ethernet-services
user@PE1# set xe-1/0/0 unit 0 encapsulation vlan-bridge
user@PE1# set xe-1/0/0 unit 0 vlan-id 10
user@PE1# set xe-1/0/0 unit 1 encapsulation vlan-bridge
user@PE1# set xe-1/0/0 unit 1 vlan-id 20
user@PE1# set xe-1/2/2 flexible-vlan-tagging
user@PE1# set xe-1/2/2 encapsulation flexible-ethernet-services
user@PE1# set xe-1/2/2 unit 0 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 0 vlan-id 10
user@PE1# set xe-1/2/2 unit 0 family bridge filter input BRI
user@PE1# set xe-1/2/2 unit 1 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 1 vlan-id 20
user@PE1# set xe-1/2/2 unit 2 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 2 vlan-id 11
user@PE1# set xe-1/2/2 unit 2 family bridge
user@PE1# set xe-1/2/2 unit 3 encapsulation vlan-bridge
user@PE1# set xe-1/2/2 unit 3 vlan-id 21
user@PE1# set xe-1/2/2 unit 3 family bridge
```

4. Configure the interfaces connecting Device PE1 with the other PE devices.

```
[edit interfaces]
user@PE1# set ge-2/1/0 gigether-options 802.3ad ae6
user@PE1# set ge-2/1/1 unit 0 family inet address 10.0.0.1/8
user@PE1# set ge-2/1/1 unit 0 family iso
user@PE1# set ge-2/1/1 unit 0 family mpls
```

5. Configure the aggregated Ethernet bundle ae6.

```
[edit interfaces]
user@PE1# set ae6 encapsulation ethernet-bridge
user@PE1# set ae6 unit 0 family bridge
```

6. Configure the loopback interface of Device PE1.

```
[edit interfaces]
user@PE1# set interfaces lo0 unit 0 family inet address 127.0.0.1/8 primary
```

7. Configure the integrated routing and bridging (IRB) interfaces for Device PE1.

```
[edit interfaces]
user@PE1# set interfaces irb unit 0 family inet address 10.0.0.1/8
user@PE1# set interfaces irb unit 0 family iso
user@PE1# set interfaces irb unit 0 family mpls
```

8. Configure the customer backbone port (CBP) interfaces on Device PE1.

```
[edit interfaces]
user@PE1# set cbp0 unit 0 family bridge interface-mode trunk
user@PE1# set cbp0 unit 0 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 0 family bridge isid-list all
user@PE1# set cbp0 unit 1 family bridge interface-mode trunk
user@PE1# set cbp0 unit 1 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 1 family bridge isid-list all
```

9. Configure the Provider Instance Port (PIP) on Device PE1.

```
[edit interfaces]
user@PE1# set pip0 unit 0 family bridge interface-mode trunk
user@PE1# set pip0 unit 0 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 0 family bridge isid-list all-service-groups
user@PE1# set pip0 unit 1 family bridge interface-mode trunk
user@PE1# set pip0 unit 1 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 1 family bridge isid-list all-service-groups
```

10. Configure the router ID and autonomous system number for Device PE1.

```
[edit routing-options]
user@PE1# set router-id 127.0.0.1
user@PE1# set autonomous-system 65221
```

11. Configure RSVP on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
```

12. Configure MPLS on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable
```

13. Configure LSPs from Device PE1 to all other PE devices.

```
[edit protocols]
user@PE1# set mpls label-switched-path PE1toPE2 from 127.0.0.1
user@PE1# set mpls label-switched-path PE1toPE2 to 127.0.0.2
user@PE1# set mpls label-switched-path PE1toPE3 from 127.0.0.1
user@PE1# set mpls label-switched-path PE1toPE3 to 127.0.0.3
```

14. Configure an internal BGP session under family EVPN from Device PE1 to all other PE devices.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp local-address 127.0.0.1
user@PE1# set bgp group ibgp family evpn signaling
user@PE1# set bgp group ibgp neighbor 127.0.0.2
user@PE1# set bgp group ibgp neighbor 127.0.0.3
```

15. Configure OSPF on all the interfaces of Device of PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf traffic-engineering
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

16. Configure a customer routing instance (I-component) on Device PE1 with type virtual switch. Assign the CBP interface, route-distinguisher, and virtual routing and forwarding (VRF) target values to the PBBN routing instance.

```
[edit routing-instances]
user@PE1# set pbbn1 instance-type virtual-switch
user@PE1# set pbbn1 interface cbp0.0
user@PE1# set pbbn1 route-distinguisher 127.0.0.1:100
user@PE1# set pbbn1 vrf-target target:100:100
```

17. Configure PBB-EVPN integration from the customer routing instance. Assign the extended I-SID list and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbbn1 protocols evpn pbb-evpn-core
user@PE1# set pbbn1 protocols evpn extended-isid-list 1000
```

```
user@PE1# set pbbn1 protocols evpn extended-isid-list 2000
user@PE1# set pbbn1 bridge-domains bda vlan-id 100
user@PE1# set pbbn1 bridge-domains bda isid-list 1000
user@PE1# set pbbn1 bridge-domains bda vlan-id-scope-local
user@PE1# set pbbn1 bridge-domains bdb vlan-id 200
user@PE1# set pbbn1 bridge-domains bdb isid-list 2000
user@PE1# set pbbn1 bridge-domains bdb vlan-id-scope-local
```

18. Configure a provider routing instance on Device PE1 with type virtual switch. Assign the PBP interface and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 instance-type virtual-switch
user@PE1# set pbn1 interface pip0.0
user@PE1# set pbn1 bridge-domains bda domain-type bridge
user@PE1# set pbn1 bridge-domains bda vlan-id 10
user@PE1# set pbn1 bridge-domains bda interface xe-1/2/2.0
user@PE1# set pbn1 bridge-domains bda interface xe-1/0/0.0
user@PE1# set pbn1 bridge-domains bdb domain-type bridge
user@PE1# set pbn1 bridge-domains bdb vlan-id 20
user@PE1# set pbn1 bridge-domains bdb interface xe-1/2/2.1
user@PE1# set pbn1 bridge-domains bdb interface xe-1/0/0.1
user@PE1# set pbn1 bridge-domains bdc domain-type bridge
user@PE1# set pbn1 bridge-domains bdc vlan-id 11
user@PE1# set pbn1 bridge-domains bdc interface xe-1/2/2.2
user@PE1# set pbn1 bridge-domains bdd domain-type bridge
user@PE1# set pbn1 bridge-domains bdd vlan-id 21
user@PE1# set pbn1 bridge-domains bdd interface xe-1/2/2.3
```

19. Configure the peer PBBN routing instance in the customer routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 pbb-options peer-instance pbbn1
```

20. Configure the service groups to be supported in the customer routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 service-groups sga service-type elan
user@PE1# set pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
user@PE1# set pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  11
user@PE1# set pbn1 service-groups sga pbb-service-options source-bmac
  00:50:50:50:50:50
user@PE1# set pbn1 service-groups sgb service-type elan
user@PE1# set pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  20
user@PE1# set pbn1 service-groups sgb pbb-service-options isid 2000 vlan-id-list
  21
```

21. Configure the bridge domains on Device PE1.

```
[edit bridge-domains]
user@PE1# set bd vlan-id none
user@PE1# set bd interface ae6.0
user@PE1# set bd routing-interface irb.0
```

Results

From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show routing-options**, **show protocols**, **show routing-instances**, and **show bridge-domains** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
   ethernet {
      device-count 16;
   }
}
network-services enhanced-ip;
```

```
user@PE1# show interfaces
xe-1/0/0 {
   flexible-vlan-tagging;
   encapsulation flexible-ethernet-services;
   unit 0 {
      encapsulation vlan-bridge;
      vlan-id 10;
   }
   unit 1 {
      encapsulation vlan-bridge;
      vlan-id 20;
   }
}
xe-1/2/2 {
   flexible-vlan-tagging;
   encapsulation flexible-ethernet-services;
   unit 0 {
      encapsulation vlan-bridge;
      vlan-id 10;
      family bridge {
         filter {
            input BRI; ## reference 'BRI' not found
         }
      }
   }
   unit 1 {
      encapsulation vlan-bridge;
      vlan-id 20;
   }
   unit 2 {
```

```
      encapsulation vlan-bridge;
      vlan-id 11;
      family bridge;
    }
    unit 3 {
      encapsulation vlan-bridge;
      vlan-id 21;
      family bridge;
    }
}
ge-2/1/0 {
    gigether-options {
      802.3ad ae6;
    }
}
ge-2/1/1 {
    unit 0 {
      family inet {
        address 10.0.0.1/8;
      }
      family iso;
      family mpls;
    }
}
ae6 {
    encapsulation ethernet-bridge;
    unit 0 {
      family bridge;
    }
}
cbp0 {
    unit 0 {
      family bridge {
        interface-mode trunk;
        bridge-domain-type bvlan;
        isid-list all;
      }
    }
    unit 1 {
      family bridge {
        interface-mode trunk;
        bridge-domain-type bvlan;
        isid-list all;
      }
    }
}
irb {
    unit 0 {
      family inet {
        address 10.0.0.1/8;
      }
      family iso;
      family mpls;
    }
}
```

```
lo0 {
   unit 0 {
      family inet {
         address 127.0.0.1/8 {
            primary;
         }
      }
   }
}
pip0 {
   unit 0 {
      family bridge {
         interface-mode trunk;
         bridge-domain-type svlan;
         isid-list all-service-groups;
      }
   }
   unit 1 {
      family bridge {
         interface-mode trunk;
         bridge-domain-type svlan;
         isid-list all-service-groups;
      }
   }
}
```

```
user@PE1# show routing-options
router-id 127.0.0.1;
autonomous-system 65221;
```

```
user@PE1# show protocols
rsvp {
   interface all;
   interface fxp0.0 {
      disable;
   }
}
mpls {
   label-switched-path PE1toPE2 {
      from 127.0.0.1;
      to 127.0.0.2;
   }
   label-switched-path PE1toPE3 {
      from 127.0.0.1;
      to 127.0.0.3;
   }
   interface all;
   interface fxp0.0 {
      disable;
   }
}
bgp {
   group ibgp {
      type internal;
```

```
    local-address 127.0.0.1;
    family evpn {
      signaling;
    }
    neighbor 127.0.0.2;
    neighbor 127.0.0.3;
  }
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface all;
    interface fxp0.0 {
      disable;
    }
  }
}
```

```
user@PE1# show routing-instances
pbbn1 {
  instance-type virtual-switch;
  interface cbp0.0;
  route-distinguisher 127.0.0.1:100;
  vrf-target target:100:100;
  protocols {
    evpn {
      pbb-evpn-core;
      extended-isid-list [ 1000 2000 ];
    }
  }
  bridge-domains {
    bda {
      vlan-id 100;
      isid-list 1000;
      vlan-id-scope-local;
    }
    bdb {
      vlan-id 200;
      isid-list 2000;
      vlan-id-scope-local;
    }
  }
}
pbn1 {
  instance-type virtual-switch;
  interface pip0.0;
  bridge-domains {
    bda {
      domain-type bridge;
      vlan-id 10;
      interface xe-1/2/2.0;
      interface xe-1/0/0.0;
    }
    bdb {
```

```
        domain-type bridge;
        vlan-id 20;
        interface xe-1/2/2.1;
        interface xe-1/0/0.1;
      }
      bdc {
        domain-type bridge;
        vlan-id 11;
        interface xe-1/2/2.2;
      }
      bdd {
        domain-type bridge;
        vlan-id 21;
        interface xe-1/2/2.3;
      }
    }
    pbb-options {
      peer-instance pbbn1;
    }
    service-groups {
      sga {
        service-type elan;
        pbb-service-options {
          isid 1000 vlan-id-list [ 10 11 ];
          source-bmac 00:50:50:50:50:50;
        }
      }
      sgb {
        service-type elan;
        pbb-service-options {
          isid 2000 vlan-id-list [ 20 21 ];
        }
      }
    }
}
```

```
user@PE1# show bridge-domains
bd {
  vlan-id none;
  interface ae6.0;
  routing-interface irb.0;
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying BGP Peering Status on page 999
- Verifying MPLS LSPs on page 999
- Verifying the EVPN Routing Instance on page 1000

Verifying BGP Peering Status

Purpose Verify that the BGP session is established between the PE devices.

Action From operational mode, run the **show bgp summary** command.

```
user@PE1> show bgp summary

Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                       8          8          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
127.0.0.2         65221          9          7       0       0        2:09 Establ

  bgp.evpn.0: 4/4/4/0
  pbbn1.evpn.0: 4/4/4/0
  __default_evpn__.evpn.0: 0/0/0/0
127.0.0.3         65221          7          7       0       0        1:25 Establ

  bgp.evpn.0: 4/4/4/0
  pbbn1.evpn.0: 4/4/4/0
__default_evpn__.evpn.0: 0/0/0/0
```

Meaning A BGP session is established between the PE devices.

Verifying MPLS LSPs

Purpose Verify the MPLS LSP status on Device PE1.

Action From operational mode, run the **show mpls lsp** command.

```
user@PE1> show mpls lsp

Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
127.0.0.2   127.0.0.1   Up     0 *                      PE1toPE2
127.0.0.3   127.0.0.1   Up     0 *                      PE1toPE3
Total 2 displayed, Up 2, Down 0

Egress LSP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
127.0.0.1   127.0.0.3   Up       0  1 FF       3        - PE3toPE1
127.0.0.1   127.0.0.2   Up       0  1 FF       3        - PE2toPE1
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions

Total 0 displayed, Up 0, Down 0
```

Verifying the EVPN Routing Instance

Purpose Verify the EVPN routing instance information.

Action From operational mode, run the **show evpn instance extensive** command.

```
user@PE1> show evpn instance extensive

Instance: __default_evpn__
  Route Distinguisher: 127.0.0.1:0
  Number of bridge domains: 0
  Number of neighbors: 0

Instance: pbbn1
  Route Distinguisher: 127.0.0.1:100
  Per-instance MAC route label: 16
  Per-instance multicast route label: 17
  PBB EVPN Core enabled
  Control word enabled
  MAC database status                       Local  Remote
    MAC advertisements:                         2       4
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
Number of local interfaces: 1 (1 up)
    Interface name  ESI                            Mode             Status
AC-Role
    cbp0.0          00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 2
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label  SG sync  IM core nexthop
          1000           0    0                Extended          Enabled   17
          Disabled
          2000           0    0                Extended          Enabled   17
          Disabled
  Number of Bundle bridge domains: 0
  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    127.0.0.2           2         0         0         2         0
    127.0.0.3           2         0         0         2         0
  Number of ethernet segments: 0
```

Meaning The output displays the **pbbn1** routing instance information, such as the integration of PBB with EVPN, the single-homed EVPN mode of operation, and the IP address of Devices PE2 and PE3 as the negihbors.

Verifying Routing Table Entries of the EVPN Routing Instance

Purpose Verify the routing table entries of the EVPN routing instance.

Action From operational mode, run the **show route table *pbbn1.evpn.0*** command.

```
user@PE1> show route table pbbn1.evpn.0

pbbn1.evpn.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2:127.0.0.1:100::1000::00:50:50:50:50:50/304 MAC/IP
                   *[EVPN/170] 00:04:20
                      Indirect
2:127.0.0.1:100::2000::00:1d:b5:a2:47:b0/304 MAC/IP
                   *[EVPN/170] 00:04:20
                      Indirect
2:127.0.0.2:100::1000::00:51:51:51:51:51/304 MAC/IP
                   *[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
2:127.0.0.2:100::2000::00:23:9c:5e:a7:b0/304 MAC/IP
                   *[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
2:127.0.0.3:100::1000::00:52:52:52:52:52/304 MAC/IP
                   *[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
                      AS path: I, validation-state: unverified
                    > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
2:127.0.0.3:100::2000::5c:5e:ab:0d:3a:b8/304 MAC/IP
                   *[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
                      AS path: I, validation-state: unverified
                    > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
3:127.0.0.1:100::1000::127.0.0.1/248 IM
                   *[EVPN/170] 00:04:20
                      Indirect
3:127.0.0.1:100::2000::127.0.0.1/248 IM
                   *[EVPN/170] 00:04:20
                      Indirect
3:127.0.0.2:100::1000::127.0.0.2/248 IM
                   *[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
3:127.0.0.2:100::2000::127.0.0.2/248 IM
                   *[BGP/170] 00:02:50, localpref 100, from 127.0.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
3:127.0.0.3:100::1000::127.0.0.3/248 IM
                   *[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
                      AS path: I, validation-state: unverified
                    > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
3:127.0.0.3:100::2000::127.0.0.3/248 IM
                   *[BGP/170] 00:02:05, localpref 100, from 127.0.0.3
                      AS path: I, validation-state: unverified
                    > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
```

Meaning The output displays the use of IRB intefraces for routing the LSPs between the PE devices.

Verifying the EVPN Database

Purpose Verify the EVPN database information on the PE devices.

Action From operational mode, run the **show evpn database** command.

```
user@PE1> show evpn database

Instance: pbbn1
VLAN  DomainId  MAC address        Active source                  Timestamp
   IP address
      2000      00:1d:b5:a2:47:b0  Local                          Apr 14 13:48:51

    2000      00:23:9c:5e:a7:b0  127.0.0.2                  Apr 14 13:53:04
      2000      5c:5e:ab:0d:3a:b8  127.0.0.3                    Apr 14 13:53:38
      1000      00:50:50:50:50:50  Local                          Apr 14 13:48:51

      1000      00:51:51:51:51:51  127.0.0.2                    Apr 14 13:53:04
      1000      00:52:52:52:52:52  127.0.0.3                    Apr 14 13:53:38
```

```
user@PE2> show evpn database

Instance: pbbn1
VLAN  DomainId  MAC address        Active source                  Timestamp
   IP address
      2000      00:1d:b5:a2:47:b0  127.0.0.1                    Apr 14 13:53:04
      2000      00:23:9c:5e:a7:b0  Local                          Apr 14 13:48:46

      2000      5c:5e:ab:0d:3a:b8  127.0.0.3                    Apr 14 13:53:37
      1000      00:50:50:50:50:50  127.0.0.1                    Apr 14 13:53:04
      1000      00:51:51:51:51:51  Local                          Apr 14 13:48:46

      1000      00:52:52:52:52:52  127.0.0.3                    Apr 14 13:53:37
```

```
user@PE3> show evpn database

Instance: pbbn1
VLAN  DomainId  MAC address        Active source                  Timestamp
   IP address
      1000      00:50:50:50:50:50  127.0.0.1                    Apr 14 13:53:34
      1000      00:51:51:51:51:51  127.0.0.2                    Apr 14 13:53:27
      1000      00:52:52:52:52:52  Local                          Apr 14 13:52:04

      2000      00:1d:b5:a2:47:b0  127.0.0.1                    Apr 14 13:53:34
      2000      00:23:9c:5e:a7:b0  127.0.0.2                    Apr 14 13:53:27
      2000      5c:5e:ab:0d:3a:b8  Local
```

Verifying the MAC Table Entries

Purpose Verify the bridge MAC table entries.

Action From operational mode, run the **show bridge mac-table** command.

```
user@PE1> show bridge mac-table

MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
    O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC, P -Pinned MAC)

Routing instance : default-switch
 Bridging domain : bd, VLAN : none
   MAC                MAC      Logical          NH     MAC
   address            flags    interface        Index  property
   00:23:9c:5e:a7:f0  D        ae6.0

MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
    O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC, P -Pinned MAC)

Routing instance : pbbn1
 Bridging domain : bda, VLAN : 100
   MAC                MAC      Logical          NH     MAC
   address            flags    interface        Index  property
   00:51:51:51:51:51  DC                        1048576
   00:52:52:52:52:52  DC                        1048581
   01:1e:83:00:03:e8  DC                        1048578

MAC flags       (S -static MAC, D -dynamic MAC, L -locally learned, C -Control
MAC
    O -OVSDB MAC, SE -Statistics enabled, NM -Non configured MAC, R -Remote PE
MAC, P -Pinned MAC)

Routing instance : pbbn1
 Bridging domain : bdb, VLAN : 200
   MAC                MAC      Logical          NH     MAC
   address            flags    interface        Index  property
   00:23:9c:5e:a7:b0  DC                        1048576
   01:1e:83:00:07:d0  DC                        1048577
   5c:5e:ab:0d:3a:b8  DC                        1048581

MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC, P -Pinned MAC)

Routing instance : pbn1
 Bridging domain : bda, ISID : 1000, VLAN : 10
   MAC                MAC      Logical                Remote
   address            flags    interface              BEB address
   00:00:00:00:0a:00  D        xe-1/0/0.0
   00:00:00:00:0a:01  D        xe-1/0/0.0
   00:00:00:00:0a:02  D        xe-1/0/0.0
   00:00:00:00:0a:03  D        xe-1/0/0.0
   00:00:00:00:0a:04  D        xe-1/0/0.0
   00:00:00:00:0a:05  D        xe-1/0/0.0
   00:00:00:00:0a:06  D        xe-1/0/0.0
   00:00:00:00:0a:07  D        xe-1/0/0.0
   00:00:00:00:0a:08  D        xe-1/0/0.0
   00:00:00:00:0a:09  D        xe-1/0/0.0
```

Meaning The output displays the MAC addresses associated with the **ae6** aggregated Ethernet bundle.

Verifying the inet.3 Routing Table Entries

Purpose Verify the inet.3 routing table entries on Device PE1.

Action From operational mode, run the **show route table inet.3** command.

```
user@PE1> show route table inet.3

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

127.0.0.2/8   *[RSVP/7/1] 00:11:15, metric 1
                    > to 10.0.0.2 via irb.0, label-switched-path PE1toPE2
127.0.0.3/8   *[RSVP/7/1] 00:09:48, metric 1
                    > to 172.17.0.1 via ge-2/1/1.0, label-switched-path PE1toPE3
```

Meaning The LSPs to Device PE2 and PE3 are routed using the IRB interface.

Related Documentation

- Provider Backbone Bridging (PBB) and EVPN Integration Overview on page 953
- Example: Configuring PBB with Multihomed EVPN on page 1005
- **pbb-evpn-core on page 1082**

Example: Configuring PBB with Multihomed EVPN

This example shows how to integrate provider backbone bridging (PBB) with Ethernet VPN (EVPN). With this integration, the control plane operations in the core are simplified, providing faster convergence and scalability enhancements than regular EVPN. The PBB-EVPN applications include Data Center Interconnect (DCI) and carrier Ethernet E-LAN services.

Requirements

This example uses the following hardware and software components:

- Four provider edge (PE) devices connected to two common multihomed customer sites, and each PE device is connected to a host.
- Two multihomed customer edge (CE) devices.
- Junos OS Release 17.2R1 or later running on all the PE routers.

Before you begin:

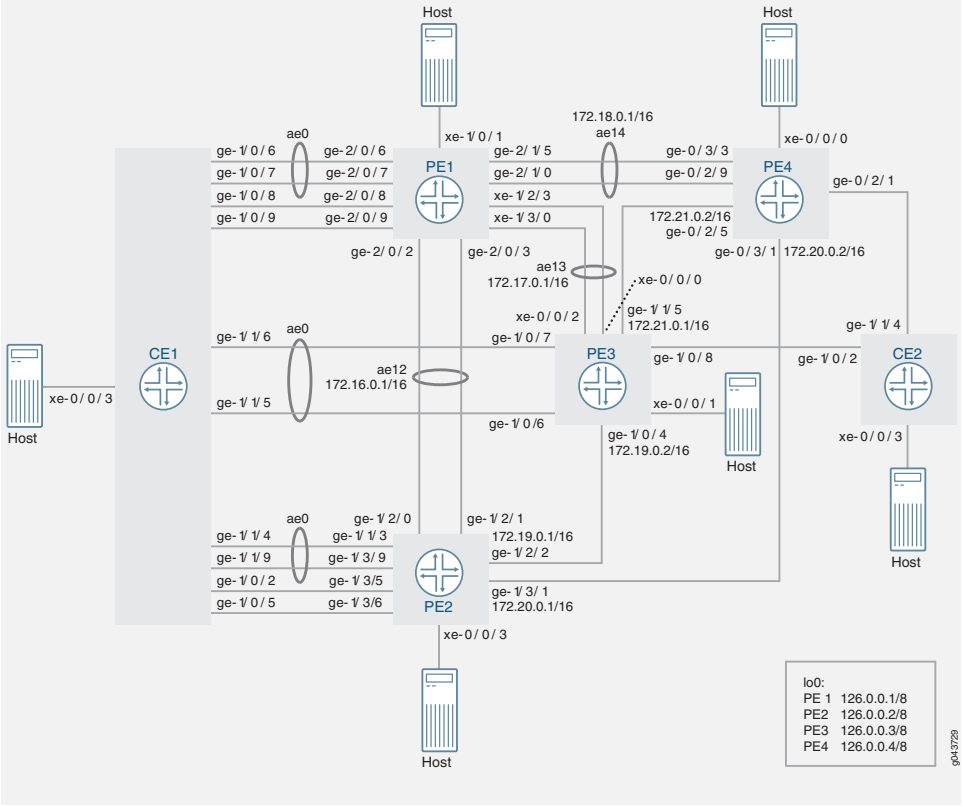
- Configure the device interfaces.
- Configure an IGP, such as OSPF, on all the PE devices.
- Establish an internal BGP session between the PE devices.
- Enable RSVP on the PE devices.
- Configure MPLS and label-switched paths (LSPs) between the PE devices.

Overview and Topology

Starting in Junos OS Release 17.2R1, PBB is integrated with Ethernet VPN (EVPN) to enable significant reduction in the control plane learning across the core, allowing a huge number of Layer 2 services, such as data center connectivity, to transit the network in a simplified manner.

In a PBB-EVPN network, the backbone core bridge (BCB) device in the PBB core is replaced with MPLS, while retaining the service scaling properties of the PBB backbone edge bridge (BEB). The B-component (provider routing instance) is signaled using EVPN BGP signaling and encapsulated inside MPLS using provider edge (PE) and provider (P) devices. Thus, PBB-EVPN combines the vast scaling property of PBB with the simplicity of a traditional basic MPLS core network, resulting in significant reduction in the amount of network-wide state information, as opposed to regular PBB.

Figure 109: PBB with Active/Standby EVPN Multihoming



In Figure 109 on page 1007, PBB is integrated with EVPN, where the CE devices are multihomed in the active/standby mode. Device CE1 is multihomed to Devices PE1, PE2, and PE3, and Device CE2 is multihomed to Device PE3 and PE4.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

CE1

```
set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 1 vlan-id 20
set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 2 vlan-id 30
set interfaces ge-1/0/2 flexible-vlan-tagging
set interfaces ge-1/0/2 encapsulation flexible-ethernet-services
```

```
set interfaces ge-1/0/2 unit 1 encapsulation vlan-bridge
set interfaces ge-1/0/2 unit 1 vlan-id 20
set interfaces ge-1/0/5 flexible-vlan-tagging
set interfaces ge-1/0/5 encapsulation flexible-ethernet-services
set interfaces ge-1/0/5 unit 2 encapsulation vlan-bridge
set interfaces ge-1/0/5 unit 2 vlan-id 30
set interfaces ge-1/0/6 gigether-options 802.3ad ae0
set interfaces ge-1/0/7 gigether-options 802.3ad ae0
set interfaces ge-1/0/8 flexible-vlan-tagging
set interfaces ge-1/0/8 encapsulation flexible-ethernet-services
set interfaces ge-1/0/8 unit 1 encapsulation vlan-bridge
set interfaces ge-1/0/8 unit 1 vlan-id 20
set interfaces ge-1/0/9 flexible-vlan-tagging
set interfaces ge-1/0/9 encapsulation flexible-ethernet-services
set interfaces ge-1/0/9 unit 2 encapsulation vlan-bridge
set interfaces ge-1/0/9 unit 2 vlan-id 30
set interfaces ge-1/1/4 gigether-options 802.3ad ae0
set interfaces ge-1/1/5 gigether-options 802.3ad ae0
set interfaces ge-1/1/6 gigether-options 802.3ad ae0
set interfaces ge-1/1/9 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ae0.0
set bridge-domains bd10 interface xe-0/0/3.0
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface xe-0/0/3.1
set bridge-domains bd20 interface ge-1/0/8.1
set bridge-domains bd20 interface ge-1/0/2.1
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface xe-0/0/3.2
set bridge-domains bd30 interface ge-1/0/9.2
set bridge-domains bd30 interface ge-1/0/5.2
```

CE2

```
set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces xe-0/0/3 unit 0 family bridge filter output f_log
set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 1 vlan-id 20
set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 2 vlan-id 30
set interfaces ge-1/0/2 flexible-vlan-tagging
set interfaces ge-1/0/2 encapsulation flexible-ethernet-services
```

```
set interfaces ge-1/0/2 unit 0 encapsulation vlan-bridge
set interfaces ge-1/0/2 unit 0 vlan-id 10
set interfaces ge-1/0/2 unit 1 encapsulation vlan-bridge
set interfaces ge-1/0/2 unit 1 vlan-id 20
set interfaces ge-1/0/2 unit 2 encapsulation vlan-bridge
set interfaces ge-1/0/2 unit 2 vlan-id 30
set interfaces ge-1/1/4 flexible-vlan-tagging
set interfaces ge-1/1/4 encapsulation flexible-ethernet-services
set interfaces ge-1/1/4 unit 0 encapsulation vlan-bridge
set interfaces ge-1/1/4 unit 0 vlan-id 10
set interfaces ge-1/1/4 unit 1 encapsulation vlan-bridge
set interfaces ge-1/1/4 unit 1 vlan-id 20
set interfaces ge-1/1/4 unit 2 encapsulation vlan-bridge
set interfaces ge-1/1/4 unit 2 vlan-id 30
set bridge-domains bd10 domain-type bridge
set bridge-domains bd10 vlan-id 10
set bridge-domains bd10 interface ge-1/1/4.0
set bridge-domains bd10 interface ge-1/0/2.0
set bridge-domains bd10 interface xe-0/0/3.0
set bridge-domains bd20 domain-type bridge
set bridge-domains bd20 vlan-id 20
set bridge-domains bd20 interface ge-1/1/4.1
set bridge-domains bd20 interface ge-1/0/2.1
set bridge-domains bd20 interface xe-0/0/3.1
set bridge-domains bd30 domain-type bridge
set bridge-domains bd30 vlan-id 30
set bridge-domains bd30 interface xe-0/0/3.2
set bridge-domains bd30 interface ge-1/1/4.2
set bridge-domains bd30 interface ge-1/0/2.2
```

PE1

```
set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-1/0/1 flexible-vlan-tagging
set interfaces xe-1/0/1 encapsulation flexible-ethernet-services
set interfaces xe-1/0/1 unit 0 encapsulation vlan-bridge
set interfaces xe-1/0/1 unit 0 vlan-id 10
set interfaces xe-1/2/3 gigether-options 802.3ad ae13
set interfaces xe-1/3/0 gigether-options 802.3ad ae13
set interfaces ge-2/0/2 gigether-options 802.3ad ae12
set interfaces ge-2/0/3 gigether-options 802.3ad ae12
set interfaces ge-2/0/6 gigether-options 802.3ad ae0
set interfaces ge-2/0/7 gigether-options 802.3ad ae0
set interfaces ge-2/0/8 flexible-vlan-tagging
set interfaces ge-2/0/8 encapsulation flexible-ethernet-services
set interfaces ge-2/0/8 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ge-2/0/8 esi single-active
set interfaces ge-2/0/8 esi source-bmac 00:22:22:22:22:22
set interfaces ge-2/0/8 unit 0 encapsulation vlan-bridge
set interfaces ge-2/0/8 unit 0 vlan-id 20
set interfaces ge-2/0/9 flexible-vlan-tagging
set interfaces ge-2/0/9 encapsulation flexible-ethernet-services
set interfaces ge-2/0/9 esi 00:33:33:33:33:33:33:33:33:33
set interfaces ge-2/0/9 esi single-active
```

```
set interfaces ge-2/0/9 esi source-bmac 00:33:33:33:33:33
set interfaces ge-2/0/9 unit 0 encapsulation vlan-bridge
set interfaces ge-2/0/9 unit 0 vlan-id 30
set interfaces ge-2/1/0 gigether-options 802.3ad ae14
set interfaces ge-2/1/5 gigether-options 802.3ad ae14
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi single-active
set interfaces ae0 esi source-bmac 00:11:11:11:11:11
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
set interfaces ae12 unit 0 family inet address 172.16.0.1/16
set interfaces ae12 unit 0 family iso
set interfaces ae12 unit 0 family mpls
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 172.17.0.1/16
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces ae14 aggregated-ether-options minimum-links 1
set interfaces ae14 unit 0 family inet address 172.18.0.1/16
set interfaces ae14 unit 0 family iso
set interfaces ae14 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.1/8 primary
set interfaces lo0 unit 0 family iso
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.1
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe1tope2 from 127.0.0.1
set protocols mpls label-switched-path pe1tope2 to 127.0.0.2
set protocols mpls label-switched-path pe1tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe1tope3 from 127.0.0.1
set protocols mpls label-switched-path pe1tope3 to 127.0.0.3
set protocols mpls label-switched-path pe1tope3 primary direct_to_pe3
```

```
set protocols mpls label-switched-path pe1tope4 from 127.0.0.1
set protocols mpls label-switched-path pe1tope4 to 127.0.0.4
set protocols mpls label-switched-path pe1tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe2 100.12.1.2 strict
set protocols mpls path direct_to_pe3 100.13.1.3 strict
set protocols mpls path direct_to_pe4 100.14.1.4 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.1
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols bgp group ibgp neighbor 127.0.0.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.1:1
set routing-instances pbbn1 vrf-target target:100:1
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface ae0.0
set routing-instances pbn1 bridge-domains bda interface xe-1/0/1.0
set routing-instances pbn1 bridge-domains bda interface ge-2/0/9.0
set routing-instances pbn1 bridge-domains bda interface ge-2/0/8.0
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
```

PE2

```
set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 flexible-vlan-tagging
set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/3 unit 0 vlan-id 10
set interfaces ge-1/2/0 gigether-options 802.3ad ae12
set interfaces ge-1/2/1 gigether-options 802.3ad ae12
set interfaces ge-1/2/2 unit 0 family inet address 172.19.0.1/16
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/3/1 unit 0 family inet address 172.20.0.1/16
set interfaces ge-1/3/1 unit 0 family iso
set interfaces ge-1/3/1 unit 0 family mpls
```

```
set interfaces ge-1/3/3 gigether-options 802.3ad ae0
set interfaces ge-1/3/5 flexible-vlan-tagging
set interfaces ge-1/3/5 encapsulation flexible-ethernet-services
set interfaces ge-1/3/5 esi 00:22:22:22:22:22:22:22:22:22
set interfaces ge-1/3/5 esi single-active
set interfaces ge-1/3/5 esi source-bmac 00:22:22:22:22:23
set interfaces ge-1/3/5 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/5 unit 0 vlan-id 20
set interfaces ge-1/3/6 flexible-vlan-tagging
set interfaces ge-1/3/6 encapsulation flexible-ethernet-services
set interfaces ge-1/3/6 esi 00:33:33:33:33:33:33:33:33:33
set interfaces ge-1/3/6 esi single-active
set interfaces ge-1/3/6 esi source-bmac 00:33:33:33:33:34
set interfaces ge-1/3/6 unit 0 encapsulation vlan-bridge
set interfaces ge-1/3/6 unit 0 vlan-id 30
set interfaces ge-1/3/9 gigether-options 802.3ad ae0
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi single-active
set interfaces ae0 esi source-bmac 00:11:11:11:11:12
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae0 unit 0 family bridge filter output f_log
set interfaces ae12 flexible-vlan-tagging
set interfaces ae12 encapsulation flexible-ethernet-services
set interfaces ae12 aggregated-ether-options minimum-links 1
set interfaces ae12 unit 0 vlan-id 1200
set interfaces ae12 unit 0 family inet address 100.12.1.2/24
set interfaces ae12 unit 0 family iso
set interfaces ae12 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.2/8 primary
set interfaces lo0 unit 0 family iso
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.2
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe2tope1 from 127.0.0.2
set protocols mpls label-switched-path pe2tope1 to 127.0.0.1
set protocols mpls label-switched-path pe2tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe2tope3 from 127.0.0.2
set protocols mpls label-switched-path pe2tope3 to 127.0.0.3
set protocols mpls label-switched-path pe2tope3 primary direct_to_pe3
```

```
set protocols mpls label-switched-path pe2tope4 from 127.0.0.2
set protocols mpls label-switched-path pe2tope4 to 127.0.0.4
set protocols mpls label-switched-path pe2tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 172.16.0.1 strict
set protocols mpls path direct_to_pe3 100.23.1.3 strict
set protocols mpls path direct_to_pe4 172.20.0.2 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.2
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols bgp group ibgp neighbor 127.0.0.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.2:1
set routing-instances pbbn1 vrf-target target:100:1
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface ae0.0
set routing-instances pbn1 bridge-domains bda interface xe-0/0/3.0
set routing-instances pbn1 bridge-domains bda interface ge-1/3/6.0
set routing-instances pbn1 bridge-domains bda interface ge-1/3/5.0
set routing-instances pbn1 bridge-domains bda bridge-options mac-statistics
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
```

PE3

```
set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/0 gigether-options 802.3ad ae13
set interfaces xe-0/0/1 flexible-vlan-tagging
set interfaces xe-0/0/1 encapsulation flexible-ethernet-services
set interfaces xe-0/0/1 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/1 unit 0 vlan-id 10
set interfaces xe-0/0/2 gigether-options 802.3ad ae13
set interfaces ge-1/0/4 unit 0 family inet address 100.23.1.3/24
set interfaces ge-1/0/4 unit 0 family iso
set interfaces ge-1/0/4 unit 0 family mpls
set interfaces ge-1/0/6 gigether-options 802.3ad ae0
set interfaces ge-1/0/7 gigether-options 802.3ad ae0
```

```
set interfaces ge-1/0/8 flexible-vlan-tagging
set interfaces ge-1/0/8 encapsulation flexible-ethernet-services
set interfaces ge-1/0/8 esi 00:44:44:44:44:44:44:44:44:44
set interfaces ge-1/0/8 esi single-active
set interfaces ge-1/0/8 esi source-bmac 00:44:44:44:44:44
set interfaces ge-1/0/8 unit 0 encapsulation vlan-bridge
set interfaces ge-1/0/8 unit 0 vlan-id 10
set interfaces ge-1/1/5 unit 0 family inet address 172.21.0.1/16
set interfaces ge-1/1/5 unit 0 family iso
set interfaces ge-1/1/5 unit 0 family mpls
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 encapsulation flexible-ethernet-services
set interfaces ae0 esi 00:11:11:11:11:11:11:11:11:11
set interfaces ae0 esi single-active
set interfaces ae0 esi source-bmac 00:11:11:11:11:13
set interfaces ae0 unit 0 encapsulation vlan-bridge
set interfaces ae0 unit 0 vlan-id 10
set interfaces ae13 flexible-vlan-tagging
set interfaces ae13 encapsulation flexible-ethernet-services
set interfaces ae13 aggregated-ether-options minimum-links 1
set interfaces ae13 unit 0 vlan-tags outer 1300
set interfaces ae13 unit 0 vlan-tags inner 13
set interfaces ae13 unit 0 family inet address 100.13.1.3/24
set interfaces ae13 unit 0 family iso
set interfaces ae13 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.3/8 primary
set interfaces lo0 unit 0 family iso
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.3
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe3tope1 from 127.0.0.3
set protocols mpls label-switched-path pe3tope1 to 127.0.0.1
set protocols mpls label-switched-path pe3tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe3tope2 from 127.0.0.3
set protocols mpls label-switched-path pe3tope2 to 127.0.0.2
set protocols mpls label-switched-path pe3tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe3tope4 from 127.0.0.3
set protocols mpls label-switched-path pe3tope4 to 127.0.0.4
set protocols mpls label-switched-path pe3tope4 primary direct_to_pe4
set protocols mpls path direct_to_pe1 172.17.0.1 strict
set protocols mpls path direct_to_pe2 172.19.0.1 strict
set protocols mpls path direct_to_pe4 172.21.0.2 strict
```

```
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.3
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.3:1
set routing-instances pbbn1 vrf-target target:100:1
set routing-instances pbbn1 protocols evpn traceoptions file pbbevpn.log
set routing-instances pbbn1 protocols evpn traceoptions file size 500m
set routing-instances pbbn1 protocols evpn traceoptions flag all
set routing-instances pbbn1 protocols evpn control-word
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface ae0.0
set routing-instances pbn1 bridge-domains bda interface xe-0/0/1.0
set routing-instances pbn1 bridge-domains bda interface ge-1/0/8.0
set routing-instances pbn1 bridge-domains bda bridge-options mac-statistics
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
```

PE4

```
set chassis aggregated-devices ethernet device-count 50
set chassis network-services enhanced-ip
set interfaces xe-0/0/0 flexible-vlan-tagging
set interfaces xe-0/0/0 encapsulation flexible-ethernet-services
set interfaces xe-0/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-0/0/0 unit 0 vlan-id 10
set interfaces ge-0/2/1 flexible-vlan-tagging
set interfaces ge-0/2/1 encapsulation flexible-ethernet-services
set interfaces ge-0/2/1 esi 00:44:44:44:44:44:44:44:44:44
set interfaces ge-0/2/1 esi single-active
set interfaces ge-0/2/1 esi source-bmac 00:44:44:44:44:45
set interfaces ge-0/2/1 unit 0 encapsulation vlan-bridge
set interfaces ge-0/2/1 unit 0 vlan-id 10
set interfaces ge-0/2/5 unit 0 family inet address 172.21.0.2/16
set interfaces ge-0/2/5 unit 0 family iso
set interfaces ge-0/2/5 unit 0 family mpls
```

```
set interfaces ge-0/2/9 gigether-options 802.3ad ae14
set interfaces ge-0/3/1 unit 0 family inet address 172.20.0.2/16
set interfaces ge-0/3/1 unit 0 family iso
set interfaces ge-0/3/1 unit 0 family mpls
set interfaces ge-0/3/3 gigether-options 802.3ad ae14
set interfaces ae14 aggregated-ether-options minimum-links 1
set interfaces ae14 unit 0 family inet address 100.14.1.4/24
set interfaces ae14 unit 0 family iso
set interfaces ae14 unit 0 family mpls
set interfaces cbp0 unit 0 family bridge interface-mode trunk
set interfaces cbp0 unit 0 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 0 family bridge isid-list all
set interfaces cbp0 unit 1 family bridge interface-mode trunk
set interfaces cbp0 unit 1 family bridge bridge-domain-type bvlan
set interfaces cbp0 unit 1 family bridge isid-list all
set interfaces lo0 unit 0 family inet address 127.0.0.4/8 primary
set interfaces pip0 unit 0 family bridge interface-mode trunk
set interfaces pip0 unit 0 family bridge bridge-domain-type svlan
set interfaces pip0 unit 0 family bridge isid-list all-service-groups
set interfaces pip0 unit 1 family bridge interface-mode trunk
set interfaces pip0 unit 1 family bridge bridge-domain-type svlan
set interfaces pip0 unit 1 family bridge isid-list all-service-groups
set routing-options router-id 127.0.0.4
set routing-options autonomous-system 65221
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path pe4tope1 from 127.0.0.4
set protocols mpls label-switched-path pe4tope1 to 127.0.0.1
set protocols mpls label-switched-path pe4tope1 primary direct_to_pe1
set protocols mpls label-switched-path pe4tope2 from 127.0.0.4
set protocols mpls label-switched-path pe4tope2 to 127.0.0.2
set protocols mpls label-switched-path pe4tope2 primary direct_to_pe2
set protocols mpls label-switched-path pe4tope3 from 127.0.0.4
set protocols mpls label-switched-path pe4tope3 to 127.0.0.3
set protocols mpls label-switched-path pe4tope3 primary direct_to_pe3
set protocols mpls path direct_to_pe1 172.18.0.1 strict
set protocols mpls path direct_to_pe2 172.20.0.1 strict
set protocols mpls path direct_to_pe3 172.21.0.1 strict
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 127.0.0.4
set protocols bgp group ibgp family evpn signaling
set protocols bgp group ibgp neighbor 127.0.0.1
set protocols bgp group ibgp neighbor 127.0.0.2
set protocols bgp group ibgp neighbor 127.0.0.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances pbbn1 instance-type virtual-switch
set routing-instances pbbn1 interface cbp0.0
set routing-instances pbbn1 route-distinguisher 127.0.0.4:1
set routing-instances pbbn1 vrf-target target:100:1
set routing-instances pbbn1 protocols evpn pbb-evpn-core
set routing-instances pbbn1 protocols evpn extended-isid-list 1000
```

```
set routing-instances pbbn1 bridge-domains bda vlan-id 100
set routing-instances pbbn1 bridge-domains bda isid-list 1000
set routing-instances pbbn1 bridge-domains bda vlan-id-scope-local
set routing-instances pbn1 instance-type virtual-switch
set routing-instances pbn1 interface pip0.0
set routing-instances pbn1 bridge-domains bda domain-type bridge
set routing-instances pbn1 bridge-domains bda vlan-id 10
set routing-instances pbn1 bridge-domains bda interface xe-0/0/0.0
set routing-instances pbn1 bridge-domains bda interface ge-0/2/1.0
set routing-instances pbn1 pbb-options peer-instance pbbn1
set routing-instances pbn1 service-groups sga service-type elan
set routing-instances pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
```

Configuring Device CE1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device CE1:

1. Set the number of aggregated Ethernet interfaces on Device CE1.

```
[edit chassis]
user@CE1# set aggregated-devices ethernet device-count 50
```

2. Set Device CE1's network services to enhanced Internet Protocol and use enhanced mode capabilities.

```
[edit chassis]
user@CE1# set chassis network-services enhanced-ip
```

3. Configure Device CE1's interfaces.

```
[edit interfaces]
user@CE1# set interfaces xe-0/0/3 flexible-vlan-tagging
user@CE1# set interfaces xe-0/0/3 encapsulation flexible-ethernet-services
user@CE1# set interfaces xe-0/0/3 unit 0 encapsulation vlan-bridge
user@CE1# set interfaces xe-0/0/3 unit 0 vlan-id 10
user@CE1# set interfaces xe-0/0/3 unit 1 encapsulation vlan-bridge
user@CE1# set interfaces xe-0/0/3 unit 1 vlan-id 20
user@CE1# set interfaces xe-0/0/3 unit 2 encapsulation vlan-bridge
user@CE1# set interfaces xe-0/0/3 unit 2 vlan-id 30
user@CE1# set interfaces ge-1/0/2 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/2 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/2 unit 1 encapsulation vlan-bridge
user@CE1# set interfaces ge-1/0/2 unit 1 vlan-id 20
user@CE1# set interfaces ge-1/0/5 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/5 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/5 unit 2 encapsulation vlan-bridge
```

```
user@CE1# set interfaces ge-1/0/5 unit 2 vlan-id 30
user@CE1# set interfaces ge-1/0/6 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/0/7 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/0/8 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/8 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/8 unit 1 encapsulation vlan-bridge
user@CE1# set interfaces ge-1/0/8 unit 1 vlan-id 20
user@CE1# set interfaces ge-1/0/9 flexible-vlan-tagging
user@CE1# set interfaces ge-1/0/9 encapsulation flexible-ethernet-services
user@CE1# set interfaces ge-1/0/9 unit 2 encapsulation vlan-bridge
user@CE1# set interfaces ge-1/0/9 unit 2 vlan-id 30
user@CE1# set interfaces ge-1/1/4 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/1/5 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/1/6 gigether-options 802.3ad ae0
user@CE1# set interfaces ge-1/1/9 gigether-options 802.3ad ae0
```

4. Configure the aggregated Ethernet bundle on Device CE1.

```
[edit interfaces]
user@CE1# set interfaces ae0 flexible-vlan-tagging
user@CE1# set interfaces ae0 encapsulation flexible-ethernet-services
user@CE1# set interfaces ae0 aggregated-ether-options minimum-links 1
user@CE1# set interfaces ae0 unit 0 encapsulation vlan-bridge
user@CE1# set interfaces ae0 unit 0 vlan-id 10
```

5. Configure the bridge domains on Device CE1.

```
[edit bridge-domains]
user@CE1# set bd10 domain-type bridge
user@CE1# set bd10 vlan-id 10
user@CE1# set bd10 interface ae0.0
user@CE1# set bd10 interface xe-0/0/3.0
user@CE1# set bd20 domain-type bridge
user@CE1# set bd20 vlan-id 20
user@CE1# set bd20 interface xe-0/0/3.1
user@CE1# set bd20 interface ge-1/0/8.1
user@CE1# set bd20 interface ge-1/0/2.1
user@CE1# set bd30 domain-type bridge
user@CE1# set bd30 vlan-id 30
user@CE1# set bd30 interface xe-0/0/3.2
user@CE1# set bd30 interface ge-1/0/9.2
user@CE1# set bd30 interface ge-1/0/5.2
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, and **show bridge-domains** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE1# show chassis
aggregated-devices {
   ethernet {
```

```
    device-count 50;
  }
}
network-services enhanced-ip;
```

```
user@CE1# show interfaces
xe-0/0/3 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 0 {
    encapsulation vlan-bridge;
    vlan-id 10;
  }
  unit 1 {
    encapsulation vlan-bridge;
    vlan-id 20;
  }
  unit 2 {
    encapsulation vlan-bridge;
    vlan-id 30;
  }
}
ge-1/0/2 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 1 {
    encapsulation vlan-bridge;
    vlan-id 20;
  }
}
ge-1/0/5 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 2 {
    encapsulation vlan-bridge;
    vlan-id 30;
  }
}
ge-1/0/6 {
  gigether-options {
    802.3ad ae0;
  }
}
ge-1/0/7 {
  gigether-options {
    802.3ad ae0;
  }
}
ge-1/0/8 {
  flexible-vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 1 {
    encapsulation vlan-bridge;
    vlan-id 20;
```

```
    }
}
ge-1/0/9 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 2 {
        encapsulation vlan-bridge;
        vlan-id 30;
    }
}
ge-1/1/4 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/5 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/6 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-1/1/9 {
    gigether-options {
        802.3ad ae0;
    }
}
ae0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    aggregated-ether-options {
        minimum-links 1;
    }
    unit 0 {
        encapsulation vlan-bridge;
        vlan-id 10;
    }
}
```

```
user@CE1# show bridge-domains
bd10 {
    domain-type bridge;
    vlan-id 10;
    interface ae0.0;
    interface xe-0/0/3.0;
}
bd20 {
    domain-type bridge;
    vlan-id 20;
    interface xe-0/0/3.1;
    interface ge-1/0/8.1;
```

```
    interface ge-1/0/2.1;
}
bd30 {
    domain-type bridge;
    vlan-id 30;
    interface xe-0/0/3.2;
    interface ge-1/0/9.2;
    interface ge-1/0/5.2;
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device PE1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device PE1:

1. Set the number of aggregated Ethernet interfaces on Device PE1.

```
[edit chassis]
user@PE1# set aggregated-devices ethernet device-count 50
```

2. Set Device PE1's network services to enhanced Internet Protocol and use enhanced mode capabilities.

```
[edit chassis]
user@PE1# set network-services enhanced-ip
```

3. Configure the CE-facing interfaces of Device PE1.

```
[edit interfaces]
user@PE1# set ge-2/0/6 gigether-options 802.3ad ae0
user@PE1# set ge-2/0/7 gigether-options 802.3ad ae0
user@PE1# set ge-2/0/8 flexible-vlan-tagging
user@PE1# set ge-2/0/8 encapsulation flexible-ethernet-services
user@PE1# set ge-2/0/8 unit 0 encapsulation vlan-bridge
user@PE1# set ge-2/0/8 unit 0 vlan-id 20
user@PE1# set ge-2/0/9 flexible-vlan-tagging
user@PE1# set ge-2/0/9 encapsulation flexible-ethernet-services
user@PE1# set ge-2/0/9 unit 0 encapsulation vlan-bridge
user@PE1# set ge-2/0/9 unit 0 vlan-id 30
```

4. Configure the aggregate Ethernet bundle on Device PE1 that connects to Device CE1.

```
[edit interfaces]
user@PE1# set ae0 flexible-vlan-tagging
```

```
user@PE1# set ae0 encapsulation flexible-ethernet-services
user@PE1# set ae0 unit 0 encapsulation vlan-bridge
user@PE1# set ae0 unit 0 vlan-id 10
```

5. Configure the EVPN multihoming parameters for the CE-facing interfaces and aggregate Ethernet bundle that connect to the multihomed customer site.

```
[edit interfaces]
user@PE1# set ge-2/0/8 esi 00:22:22:22:22:22:22:22:22:22
user@PE1# set ge-2/0/8 esi single-active
user@PE1# set ge-2/0/8 esi source-bmac 00:22:22:22:22:22
user@PE1# set ge-2/0/9 esi 00:33:33:33:33:33:33:33:33:33
user@PE1# set ge-2/0/9 esi single-active
user@PE1# set ge-2/0/9 esi source-bmac 00:33:33:33:33:33
user@PE1# set ae0 esi 00:11:11:11:11:11:11:11:11:11
user@PE1# set ae0 esi single-active
user@PE1# set ae0 esi source-bmac 00:11:11:11:11:11
```

NOTE: In this example, the EVPN multihoming is operating in the active/standby mode. To configure active/active EVPN multihoming, include the **active-active** statement at the **[edit interfaces** *interface-name* **esi]** hierarchy level instead of the **single-active** statement.

6. Configure the interface of Device PE1 that connects to Devices PE2, PE3, and PE4.

```
[edit interfaces]
user@PE1# set ge-2/0/2 gigether-options 802.3ad ae12
user@PE1# set ge-2/0/3 gigether-options 802.3ad ae12
user@PE1# set xe-1/2/3 gigether-options 802.3ad ae13
user@PE1# set xe-1/3/0 gigether-options 802.3ad ae13
user@PE1# set ge-2/1/0 gigether-options 802.3ad ae14
user@PE1# set ge-2/1/5 gigether-options 802.3ad ae14
```

7. Configure the aggregate Ethernet bundle on Device PE1 that connect to Devices PE2, PE3, and PE4.

```
[edit interfaces]
user@PE1# set ae12 flexible-vlan-tagging
user@PE1# set ae12 encapsulation flexible-ethernet-services
user@PE1# set ae12 aggregated-ether-options minimum-links 1
user@PE1# set ae12 unit 0 vlan-id 1200
user@PE1# set ae12 unit 0 family inet address 172.16.0.1/16
user@PE1# set ae12 unit 0 family iso
user@PE1# set ae12 unit 0 family mpls
user@PE1# set ae13 flexible-vlan-tagging
user@PE1# set ae13 encapsulation flexible-ethernet-services
user@PE1# set ae13 aggregated-ether-options minimum-links 1
user@PE1# set ae13 unit 0 vlan-tags outer 1300
```

```
user@PE1# set ae13 unit 0 vlan-tags inner 13
user@PE1# set ae13 unit 0 family inet address 172.17.0.1/16
user@PE1# set ae13 unit 0 family iso
user@PE1# set ae13 unit 0 family mpls
user@PE1# set ae14 aggregated-ether-options minimum-links 1
user@PE1# set ae14 unit 0 family inet address 172.18.0.1/16
user@PE1# set ae14 unit 0 family iso
user@PE1# set ae14 unit 0 family mpls
```

8. Configure the interface of Device PE1 that connects to the host.

```
[edit interfaces]
user@PE1# set xe-1/0/1 flexible-vlan-tagging
user@PE1# set xe-1/0/1 encapsulation flexible-ethernet-services
user@PE1# set xe-1/0/1 unit 0 encapsulation vlan-bridge
user@PE1# set xe-1/0/1 unit 0 vlan-id 10
```

9. Configure the loopback interface of Device PE1.

```
[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 127.0.0.1/8 primary
user@PE1# set lo0 unit 0 family iso
```

10. Configure the customer backbone port (CBP) interfaces on Device PE1.

```
[edit interfaces]
user@PE1# set cbp0 unit 0 family bridge interface-mode trunk
user@PE1# set cbp0 unit 0 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 0 family bridge isid-list all
user@PE1# set cbp0 unit 1 family bridge interface-mode trunk
user@PE1# set cbp0 unit 1 family bridge bridge-domain-type bvlan
user@PE1# set cbp0 unit 1 family bridge isid-list all
```

11. Configure the Provider Instance Port (PIP) on Device PE1.

```
[edit interfaces]
user@PE1# set pip0 unit 0 family bridge interface-mode trunk
user@PE1# set pip0 unit 0 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 0 family bridge isid-list all-service-groups
user@PE1# set pip0 unit 1 family bridge interface-mode trunk
user@PE1# set pip0 unit 1 family bridge bridge-domain-type svlan
user@PE1# set pip0 unit 1 family bridge isid-list all-service-groups
```

12. Configure the router ID and autonomous system number for Device PE1.

```
[edit routing-options]
user@PE1# set router-id 127.0.0.1
user@PE1# set autonomous-system 65221
```

13. Configure RSVP on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set rsvp interface all
user@PE1# set rsvp interface fxp0.0 disable
```

14. Configure MPLS on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set mpls interface fxp0.0 disable
```

15. Configure LSPs from Device PE1 to all other PE devices.

```
[edit protocols]
user@PE1# set mpls label-switched-path pe1tope2 from 127.0.0.1
user@PE1# set mpls label-switched-path pe1tope2 to 127.0.0.2
user@PE1# set mpls label-switched-path pe1tope2 primary direct_to_pe2
user@PE1# set mpls label-switched-path pe1tope3 from 127.0.0.1
user@PE1# set mpls label-switched-path pe1tope3 to 127.0.0.3
user@PE1# set mpls label-switched-path pe1tope3 primary direct_to_pe3
user@PE1# set mpls label-switched-path pe1tope4 from 127.0.0.1
user@PE1# set mpls label-switched-path pe1tope4 to 127.0.0.4
user@PE1# set mpls label-switched-path pe1tope4 primary direct_to_pe4
```

16. Configure MPLS paths from Device PE1 to all other PE devices.

```
[edit protocols]
user@PE1# set mpls path direct_to_pe2 100.12.1.2 strict
user@PE1# set mpls path direct_to_pe3 100.13.1.3 strict
user@PE1# set mpls path direct_to_pe4 100.14.1.4 strict
```

17. Configure an internal BGP session under family EVPN from Device PE1 to all other PE devices.

```
[edit protocols]
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp local-address 127.0.0.1
user@PE1# set bgp group ibgp family evpn signaling
user@PE1# set bgp group ibgp neighbor 127.0.0.2
user@PE1# set bgp group ibgp neighbor 127.0.0.3
user@PE1# set bgp group ibgp neighbor 127.0.0.4
```

18. Configure OSPF on all the interfaces of Device of PE1, excluding the management interface.

```
[edit protocols]
```

```
user@PE1# set ospf traffic-engineering
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
```

19. Configure a customer routing instance (I-component) on Device PE1 with type virtual switch. Assign the CBP interface, route-distinguisher, and virtual routing and forwarding (VRF) target values to the PBBN routing instance.

```
[edit routing-instances]
user@PE1# set pbbn1 instance-type virtual-switch
user@PE1# set pbbn1 interface cbp0.0
user@PE1# set pbbn1 route-distinguisher 127.0.0.1:1
user@PE1# set pbbn1 vrf-target target:100:1
```

20. Configure PBB-EVPN integration from the customer routing instance. Assign the extended I-SID list and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbbn1 protocols evpn pbb-evpn-core
user@PE1# set pbbn1 protocols evpn extended-isid-list 1000
user@PE1# set pbbn1 bridge-domains bda vlan-id 100
user@PE1# set pbbn1 bridge-domains bda isid-list 1000
user@PE1# set pbbn1 bridge-domains bda vlan-id-scope-local
```

21. Configure a provider routing instance on Device PE1 with type virtual switch. Assign the PBP interface and bridge domains to the routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 instance-type virtual-switch
user@PE1# set pbn1 interface pip0.0
user@PE1# set pbn1 bridge-domains bda domain-type bridge
user@PE1# set pbn1 bridge-domains bda vlan-id 10
user@PE1# set pbn1 bridge-domains bda interface ae0.0
user@PE1# set pbn1 bridge-domains bda interface xe-1/0/1.0
user@PE1# set pbn1 bridge-domains bda interface ge-2/0/9.0
user@PE1# set pbn1 bridge-domains bda interface ge-2/0/8.0
```

22. Configure the peer PBBN routing instance in the customer routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 pbb-options peer-instance pbbn1
```

23. Configure the service groups to be supported in the customer routing instance.

```
[edit routing-instances]
user@PE1# set pbn1 service-groups sga service-type elan
user@PE1# set pbn1 service-groups sga pbb-service-options isid 1000 vlan-id-list
  10
```

Results From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **routing-options**, **show protocols** and **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
aggregated-devices {
   ethernet {
      device-count 50;
   }
}
network-services enhanced-ip;
```

```
user@PE1# show interfaces
xe-1/0/1 {
   flexible-vlan-tagging;
   encapsulation flexible-ethernet-services;
   unit 0 {
      encapsulation vlan-bridge;
      vlan-id 10;
   }
}
xe-1/2/3 {
   gigether-options {
      802.3ad ae13;
   }
}
xe-1/3/0 {
   gigether-options {
      802.3ad ae13;
   }
}
ge-2/0/2 {
   gigether-options {
      802.3ad ae12;
   }
}
ge-2/0/3 {
   gigether-options {
      802.3ad ae12;
   }
}
ge-2/0/6 {
   gigether-options {
      802.3ad ae0;
   }
}
ge-2/0/7 {
   gigether-options {
      802.3ad ae0;
   }
}
ge-2/0/8 {
   flexible-vlan-tagging;
```

```
    encapsulation flexible-ethernet-services;
    esi {
      00:22:22:22:22:22:22:22:22:22;
      single-active;
      source-bmac 00:22:22:22:22:22;
    }
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id 20;
    }
  }
  ge-2/0/9 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:33:33:33:33:33:33:33:33:33;
      single-active;
      source-bmac 00:33:33:33:33:33;
    }
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id 30;
    }
  }
  ge-2/1/0 {
    gigether-options {
      802.3ad ae14;
    }
  }
  ge-2/1/5 {
    gigether-options {
      802.3ad ae14;
    }
  }
  ae0 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    esi {
      00:11:11:11:11:11:11:11:11:11;
      single-active;
      source-bmac 00:11:11:11:11:11;
    }
    unit 0 {
      encapsulation vlan-bridge;
      vlan-id 10;
    }
  }
  ae12 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    aggregated-ether-options {
      minimum-links 1;
    }
    unit 0 {
      vlan-id 1200;
```

```
      family inet {
        address 172.16.0.1/16;
      }
      family iso;
      family mpls;
    }
  }
  ae13 {
    flexible-vlan-tagging;
    encapsulation flexible-ethernet-services;
    aggregated-ether-options {
      minimum-links 1;
    }
    unit 0 {
      vlan-tags outer 1300 inner 13;
      family inet {
        address 172.17.0.1/16;
      }
      family iso;
      family mpls;
    }
  }
  ae14 {
    aggregated-ether-options {
      minimum-links 1;
    }
    unit 0 {
      family inet {
        address 172.18.0.1/16;
      }
      family iso;
      family mpls;
    }
  }
  cbp0 {
    unit 0 {
      family bridge {
        interface-mode trunk;
        bridge-domain-type bvlan;
        isid-list all;
      }
    }
    unit 1 {
      family bridge {
        interface-mode trunk;
        bridge-domain-type bvlan;
        isid-list all;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 127.0.0.1/8 {
          primary;
```

```
        }
      }
      family iso;
    }
}
pip0 {
    unit 0 {
      family bridge {
        interface-mode trunk;
        bridge-domain-type svlan;
        isid-list all-service-groups;
      }
    }
    unit 1 {
      family bridge {
        interface-mode trunk;
        bridge-domain-type svlan;
        isid-list all-service-groups;
      }
    }
}
```

```
user@PE1# show routing-options
router-id 127.0.0.1;
autonomous-system 65221;
```

```
user@PE1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
      disable;
    }
}
mpls {
    label-switched-path pe1tope2 {
      from 127.0.0.1;
      to 127.0.0.2;
      primary direct_to_pe2;
    }
    label-switched-path pe1tope3 {
      from 127.0.0.1;
      to 127.0.0.3;
      primary direct_to_pe3;
    }
    label-switched-path pe1tope4 {
      from 127.0.0.1;
      to 127.0.0.4;
      primary direct_to_pe4;
    }
    path direct_to_pe2 {
      100.12.1.2 strict;
    }
    path direct_to_pe3 {
      100.13.1.3 strict;
```

```
    }
    path direct_to_pe4 {
      100.14.1.4 strict;
    }
    interface all;
    interface fxp0.0 {
      disable;
    }
  }
  bgp {
    group ibgp {
      type internal;
      local-address 127.0.0.1;
      family evpn {
        signaling;
      }
      neighbor 127.0.0.2;
      neighbor 127.0.0.3;
      neighbor 127.0.0.4;
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
  }
```

```
user@PE1# show routing-instances
pbbn1 {
    instance-type virtual-switch;
    interface cbp0.0;
    route-distinguisher 127.0.0.1:1;
    vrf-target target:100:1;
    protocols {
        evpn {
            pbb-evpn-core;
            extended-isid-list 1000;
        }
    }
    bridge-domains {
        bda {
            vlan-id 100;
            isid-list 1000;
            vlan-id-scope-local;
        }
    }
}
pbn1 {
    instance-type virtual-switch;
    interface pip0.0;
```

```
    bridge-domains {
      bda {
        domain-type bridge;
        vlan-id 10;
        interface ae0.0;
        interface xe-1/0/1.0;
        interface ge-2/0/9.0;
        interface ge-2/0/8.0;
      }
    }
    pbb-options {
      peer-instance pbbn1;
    }
    service-groups {
      sga {
        service-type elan;
        pbb-service-options {
          isid 1000 vlan-id-list 10;
        }
      }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying BGP Peering Status

Purpose Verify that the BGP session is established between the PE devices.

Action From operational mode, run the **show bgp summary** command.

```
user@PE1> show bgp summary

Groups: 1 Peers: 3 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                      13         13          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
127.0.0.2         65221         10          8       0       0          42 Establ

  bgp.evpn.0: 6/6/6/0
  pbbn1.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 3/3/3/0
127.0.0.3         65221          8          8       0       0          40 Establ

  bgp.evpn.0: 4/4/4/0
  pbbn1.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 1/1/1/0
127.0.0.4         65221          9         11       0       0        1:04 Establ

  bgp.evpn.0: 3/3/3/0
  pbbn1.evpn.0: 3/3/3/0
  __default_evpn__.evpn.0: 0/0/0/0
```

Meaning A BGP session is established between the PE devices.

Verify MAC Table Entries

Purpose Verify the number of **rbeb** interfaces learned in the MAC table on all PEs.

Action From operational mode, run the **show bgp summary** command.

```
user@PE1> show bridge mac-table count instance pbn1 bridge-domain bda

10 MAC address learned in routing instance pbn1 bridge domain bda

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ae0.0:10                         0
    ge-2/0/8.0:10                   10
    ge-2/0/9.0:10                    0
    rbeb.32772                       0
    rbeb.32771                       0
    xe-1/0/0.0:10                    0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
               10                    10
```

Meaning For VLAN ID 10, 10 MAC addresses have been learned in the MAC table of Device PE1.

Verifying the EVPN Database

Purpose Verify the EVPN database information on Device PE4.

Action From operational mode, run the **show evpn database** command.

```
user@PE1> show evpn database

Instance: pbbn1
VLAN  DomainId  MAC address        Active source                  Timestamp
   IP address
      1000      00:11:11:11:11:11  127.0.0.1                  Jan 26 12:09:29
      1000      00:11:11:11:11:12  127.0.0.2                  Jan 26 12:09:38
      1000      00:1d:b5:a2:47:b0  127.0.0.1                  Jan 26 12:09:10
      1000      00:22:22:22:22:22  127.0.0.1                  Jan 26 12:09:29
      1000      00:23:9c:5e:a7:b0  Local                         Jan 26 12:08:02

      1000      00:23:9c:f0:f9:b0  127.0.0.3                  Jan 26 12:09:30
      1000      00:33:33:33:33:33  127.0.0.1                  Jan 26 12:09:29
      1000      00:44:44:44:44:44  127.0.0.3                  Jan 26 12:09:34
      1000      00:44:44:44:44:45  Local                         Jan 26 12:09:28

      1000      80:71:1f:c1:ed:b0  127.0.0.2                  Jan 26 12:09:31
```

Verifing EVPN Routing Instances

Purpose Verify the EVPN routing instance information on all the PE devices.

Action From operational mode, run the **show evpn routing-instance** command.

```
user@PE1> show evpn routing-instance

Instance: pbbn1
  Route Distinguisher: 127.0.0.1:1
  Per-instance MAC route label: 300080
  Per-instance multicast route label: 300096
  PBB EVPN Core enabled
  Control word enabled
  MAC database status                       Local  Remote
    MAC advertisements:                         4       6
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 4 (4 up)
    Interface name  ESI                               Mode             Status
AC-Role
    ae0.0           00:11:11:11:11:11:11:11:11:11  single-active    Up
Root
    cbp0.0          00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
    ge-2/0/8.0      00:22:22:22:22:22:22:22:22:22  single-active    Up
Root
    ge-2/0/9.0      00:33:33:33:33:33:33:33:33:33  single-active    Up
Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label  SG sync  IM core nexthop
          1000           0    0                Extended         Enabled   300096
          Disabled
  Number of Bundle bridge domains: 0
  Number of neighbors: 3
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    127.0.0.2           2         0         0         1         0
    127.0.0.3           2         0         0         1         0
    127.0.0.4           2         0         0         1         0
  Number of ethernet segments: 3
    ESI: 00:11:11:11:11:11:11:11:11:11
      Status: Resolved by IFL ae0.0
      Local interface: ae0.0, Status: Up/Blocking
      Designated forwarder: 127.0.0.2
      Backup forwarder: 127.0.0.1
      Backup forwarder: 127.0.0.3
      Last designated forwarder update: Jan 26 12:43:42
      Minimum ISID: 1000
    ESI: 00:22:22:22:22:22:22:22:22:22
      Status: Resolved by IFL ge-2/0/8.0
      Local interface: ge-2/0/8.0, Status: Up/Forwarding
      Designated forwarder: 127.0.0.1
      Backup forwarder: 127.0.0.2
      Last designated forwarder update: Jan 26 12:43:42
      Minimum ISID: 1000
    ESI: 00:33:33:33:33:33:33:33:33:33
      Status: Resolved by IFL ge-2/0/9.0
      Local interface: ge-2/0/9.0, Status: Up/Forwarding
      Designated forwarder: 127.0.0.1
      Backup forwarder: 127.0.0.2
```

```
    Last designated forwarder update: Jan 26 12:43:42
    Minimum ISID: 1000
```

```
user@PE2> show evpn routing-instance
```

```
Instance: pbbn1
  Route Distinguisher: 127.0.0.2:1
  Per-instance MAC route label: 338721
  Per-instance multicast route label: 338737
  PBB EVPN Core enabled
  Control word enabled
  MAC database status                       Local  Remote
    MAC advertisements:                         2       8
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 4 (4 up)
    Interface name  ESI                            Mode             Status
AC-Role
    ae0.0           00:11:11:11:11:11:11:11:11:11  single-active    Up
Root
    cbp0.0          00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
    ge-1/3/5.0      00:22:22:22:22:22:22:22:22:22  single-active    Up
Root
    ge-1/3/6.0      00:33:33:33:33:33:33:33:33:33  single-active    Up
Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label  SG sync  IM core nexthop
          1000           0    0                Extended         Enabled   338737
          Disabled
  Number of Bundle bridge domains: 0
  Number of neighbors: 3
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    127.0.0.1           4         0         0         1         0
    127.0.0.3           2         0         0         1         0
    127.0.0.4           2         0         0         1         0
  Number of ethernet segments: 3
    ESI: 00:11:11:11:11:11:11:11:11:11
      Status: Resolved by IFL ae0.0
      Local interface: ae0.0, Status: Up/Forwarding
      Designated forwarder: 127.0.0.2
      Backup forwarder: 127.0.0.1
      Backup forwarder: 127.0.0.3
      Last designated forwarder update: Jan 26 12:09:37
      Minimum ISID: 1000
    ESI: 00:22:22:22:22:22:22:22:22:22
      Status: Resolved by IFL ge-1/3/5.0
      Local interface: ge-1/3/5.0, Status: Up/Blocking
      Designated forwarder: 127.0.0.1
      Backup forwarder: 127.0.0.2
      Last designated forwarder update: Jan 26 12:09:33
      Minimum ISID: 1000
    ESI: 00:33:33:33:33:33:33:33:33:33
      Status: Resolved by IFL ge-1/3/6.0
```

```
    Local interface: ge-1/3/6.0, Status: Up/Blocking
    Designated forwarder: 127.0.0.1
    Backup forwarder: 127.0.0.2
    Last designated forwarder update: Jan 26 12:09:33
    Minimum ISID: 1000
```

user@PE3> show evpn routing-instance

```
Instance: pbbn1
  Route Distinguisher: 127.0.0.3:1
  Per-instance MAC route label: 338833
  Per-instance multicast route label: 338849
  PBB EVPN Core enabled
  Control word enabled
  MAC database status                       Local  Remote
    MAC advertisements:                         2       8
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 3 (3 up)
    Interface name  ESI                               Mode             Status
AC-Role
    ae0.0           00:11:11:11:11:11:11:11:11:11  single-active    Up
Root
    cbp0.0          00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
    ge-1/0/8.0      00:44:44:44:44:44:44:44:44:44  single-active    Up
Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label  SG sync  IM core nexthop
          1000           0    0                Extended         Enabled   338849
          Disabled
  Number of Bundle bridge domains: 0
  Number of neighbors: 3
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    127.0.0.1           4         0         0         1         0
    127.0.0.2           2         0         0         1         0
    127.0.0.4           2         0         0         1         0
  Number of ethernet segments: 2
    ESI: 00:11:11:11:11:11:11:11:11:11
      Status: Resolved by IFL ae0.0
      Local interface: ae0.0, Status: Up/Blocking
      Designated forwarder: 127.0.0.2
      Backup forwarder: 127.0.0.1
      Backup forwarder: 127.0.0.3
      Last designated forwarder update: Jan 26 12:09:46
      Minimum ISID: 1000
    ESI: 00:44:44:44:44:44:44:44:44:44
      Status: Resolved by IFL ge-1/0/8.0
      Local interface: ge-1/0/8.0, Status: Up/Forwarding
      Designated forwarder: 127.0.0.3
      Backup forwarder: 127.0.0.4
      Last designated forwarder update: Jan 26 12:09:31
      Minimum ISID: 1000
```

```
user@PE4> show evpn routing-instance
```

```
Instance: pbbn1
  Route Distinguisher: 127.0.0.4:1
  Per-instance MAC route label: 301520
  Per-instance multicast route label: 301536
  PBB EVPN Core enabled
  Control word enabled
  MAC database status                       Local  Remote
    MAC advertisements:                         2       8
    MAC+IP advertisements:                      0       0
    Default gateway MAC advertisements:         0       0
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                               Mode             Status
AC-Role
    cbp0.0          00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
    ge-0/2/1.0      00:44:44:44:44:44:44:44:44:44  single-active    Up
Root
  Number of IRB interfaces: 0 (0 up)
  Number of bridge domains: 1
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label  SG sync  IM core nexthop
          1000           0    0                Extended         Enabled   301536
          Disabled
  Number of Bundle bridge domains: 0
  Number of neighbors: 3
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label

    127.0.0.1           4         0         0         1         0
    127.0.0.2           2         0         0         1         0
    127.0.0.3           2         0         0         1         0
  Number of ethernet segments: 1
    ESI: 00:44:44:44:44:44:44:44:44:44
      Status: Resolved by IFL ge-0/2/1.0
      Local interface: ge-0/2/1.0, Status: Up/Blocking
      Designated forwarder: 127.0.0.3
      Backup forwarder: 127.0.0.4
      Last designated forwarder update: Jan 26 12:43:52
      Minimum ISID: 1000
```

Meaning The command output displays PBB-EVPN integration, where EVPN is in the active/standby multihoming mode. If EVPN multihoming was configured in the active/active mode, the status of all the ESIs would be **Up/Forwarding**. In the active/standby multihoming mode, one ESI is in the **Up/Forwarding** state and all other ESIs remain in the **Up/Blocking** state.

Related Documentation

- Provider Backbone Bridging (PBB) and EVPN Integration Overview on page 953
- Example: Configuring PBB with Single-Homed EVPN on page 983
- pbb-evpn-core on page 1082

CHAPTER 28

Configuring MAC Pinning for PBB-EVPNs

PBB-EVPN MAC Pinning Overview

Starting in Junos OS Release 17.2, the MAC pinning feature is enabled on provider backbone bridging (PBB) and Ethernet VPN (EVPN) integration, including customer edge (CE) interfaces and EVPN over PBB core in both all-active or single-active mode.

The MAC pinning feature is used to avoid loops in a network and is also used for MAC security restriction by avoiding MAC move on duplicate MAC detection. When MAC pinning is enabled, the dynamically learned MAC addresses are not allowed to move to any other interface in a bridge domain until it is aged out and traffic received with the same source MAC address on other bridge interfaces are discarded. This feature is an advantage over blocking of the complete interface on duplicate MAC detection or loop, as MAC pinning works at the MAC label. This feature is local to a provider edge (PE) device and does not require any interoperability.

PBB has I-component and B-Component, where I-component (customer routing instance) is responsible for mapping the CE port traffic to the instance source ID (I-SID), and the B-component learns and forwards traffic on the backbone port. Traffic received from the MPLS core or from the PBB port is classified and based on the I-SID and PBB MAC, and gets mapped to the correct I-component. Remote customer MAC addresses are learned over remote backbone edge port (BEB) interface in the I-component bridge domain. This interface is created dynamically on PBB neighbor detection. MAC addresses learned over the remote BEB interface in I-component are pinned when MAC pinning is enabled for PBB-EVPN.

To configure MAC pining for PBB-EVPN, include the **mac-pinning** statement at the **[edit routing-instances *pbbn* protocols evpn]**, where **pbbn** is the PBB routing instance over backbone port (B-component). With this configuration, the dynamically learned MAC addresses in the PBB I-component bridge domain over CE interfaces, as well as PBB-MPLS core interfaces are pinned.

When configuring the PBB-EVPN MAC pinning feature, take the following into consideration:

- PBB-EVPN MAC pinning is supported on MX Series routers with MPC and MIC interfaces only.
- PBB-EVPN MAC pinning is supported on Ethernet Layer 2 bridge interfaces only.
- When there is a MAC move between the I-component and an access interface, the MAC address is learned locally over the PBB-EVPN MPLS core over a remote BEB interface in the I-component bridge domain. The MAC moves between the CE or core interfaces for this MAC is not allowed.
- In MAC pinning for PBB with EVPN active-active and single-active multihoming, MAC pinning must be enabled or disabled on all the multihomed PE devices in the broadcast domain. This is because MAC pining at a multihomed PE device is local to the PE, and it is possible that a MAC address that is pinned towards a multihomed CE device and PE device is also pinned toward a single-homed customer site or toward any other Ethernet segment identifier (ESI) at another multihomed PE device.
- A next hop bridge domain is created in PBB-EVPN I-component bridge domain toward the B-component when there is an unresolved source MAC notification when the first remote MAC address is received. As a result, the first MAC address learned over PBB back bone core interface can be delayed on pinning, and may result moving to other single-homed or ESI interface if the same MAC traffic is received.
- Static MAC addresses are given preference over dynamic pin MACs.
- MAC pinning is enabled for all neighbors of a PBB routing instance and cannot be enabled for a specific neighbor.
- PBB-EVPN MAC pin discard notification is not generated for a remote BEB interface when traffic is discarded due to MAC pinning until a MAC is learned locally over the remote BEB interface.

Related Documentation

- *Understanding MAC Pinning*
- Configuring PBB-EVPN MAC Pinning on page 1041
- *mac-pinning*
- pbb-evpn-core on page 1082

Configuring PBB-EVPN MAC Pinning

Starting in Junos OS Release 17.2, the MAC pinning feature is enabled on provider backbone bridging (PBB) and Ethernet VPN (EVPN) integration, including customer edge (CE) interfaces and EVPN over PBB core in both all-active or single-active mode.

When MAC pinning is enabled, the dynamically learned MAC addresses are not allowed to move to any other interface in a bridge domain until it is aged out and traffic received with the same source MAC address on other bridge interfaces are discarded. This feature is an advantage over blocking of the complete interface on duplicate MAC detection or loop detection, as MAC pinning works at the MAC label. This feature is local to a provider edge (PE) device and does not require any interoperability.

Before you begin:

- Configure the device interfaces, including the customer backbone port (CBP) interface, the provider instance port (PIP) interfaces, and the loopback interface. Assign the bridge family to the interfaces.
- Assign the router ID and autonomous system ID to the device.
- Configure an internal BGP group with EVPN signaling.
- Enable the following protocols on the device:
  - MPLS
  - LDP
  - OSPF

To enable MAC pinning on PBB-EVPN:

1. Configure the B-component routing instance.

   Assign the virtual switch instance type and the CBP interface to it. Configure other routing instance attributes like route distinguisher and virtual routing and forwarding (VRF) target to the routing instance.

   NOTE: Configure B-component routing instances for other CBP interface units on the device, and assign different VLAN IDs and I-SID lists for the different interface units.

   ```
   [edit routing-instances]
   user@R1# set pbbn instance-type virtual-switch
   user@R1# set pbbn interface cbp-interface
   user@R1# set pbbn route-distinguisher route-distinguisher-value
   user@R1# set pbbn vrf-target vrf-target
   ```

2. Enable PBB- EVPN integration for the B-component routing instance.

```
[edit routing-instances]
user@R1# set pbbn protocols evpn pbb-evpn-core
```

3. Enable MAC pinning for the B-component routing instance.

```
[edit routing-instances]
user@R1# set pbbn protocols evpn mac-pinning
```

4. Assign instance source IDs (I-SID) list to the B-component routing instance.

```
[edit routing-instances]
user@R1# set pbbn protocols evpn extended-isid-list extended-isid-list
```

5. Configure a bridge domain for the B-component routing instance and assign a VLAN and and I-SID list to the bridge domain.

```
[edit routing-instances]
user@R1# set pbbn bridge-domains bridge-domain vlan-id vlan-id
user@R1# set pbbn bridge-domains bridge-domain isid-list isid-list
```

6. Configure the I-component routing instance.

   Assign the virtual switch instance type and the PIP interface to it.

   NOTE: Configure I-component routing instances for other PIP interface units on the device, and assign different bridge domains, VLAN IDs and I-SID lists for the different interface units.

```
[edit routing-instances]
user@R1# set pbn instance-type virtual-switch
user@R1# set pbn interface pip-interface
```

7. Configure bridge domain for the I-component routing instance and assign interfaces and VLANs to the bridge domain.

```
[edit routing-instances]
user@R1# set pbn bridge-domains bridge-domain domain-type bridge
user@R1# set pbn bridge-domains bridge-domain vlan-id vlan-id
user@R1# set pbn bridge-domains bridge-domain interface interface-name
```

8. Enable MAC pinning for the interface in the I-component routing instance.

```
[edit routing-instances]
user@R1# set pbn bridge-domains bridge-domain bridge-options interface
  interface-name mac-pinning
```

9. Configure peering between the B-component and the I-component routing instances.

```
[edit routing-instances]
user@R1# set pbn bridge-domains bridge-domain pbb-options peer-instance pbbn
```

10. Configure PBB service group and assign I-SID and VLAN ID list.

```
[edit routing-instances]
user@R1# set pbn service-groups service-group pbb-service-options isid isid vlan-id-list
  valn-id-list
```

Related Documentation

- *Understanding MAC Pinning*
- PBB-EVPN MAC Pinning Overview on page 1039
- *mac-pinning*
- pbb-evpn-core on page 1082

PART 8

Configuration Statements and Operational Commands

CHAPTER 29

EVPN Configuration Statements

- vlan-id (routing instance) on page 1089
- vpws-service-id on page 1090
- vrf-export on page 1091
- vrf-import on page 1092
- vrf-target on page 1093

advertise-ip-prefix

Syntax

```
advertise-ip-prefix
```

Hierarchy Level

```
[edit protocols evpn]
[edit routing-instances routing-instance-name protocols evpn],
```

Release Information

Statement introduced in Junos OS Release 17.1 on MX Series routers with MPCs.

Description

Enable the advertisement of Ethernet VPN (EVPN) pure type-5 routes for the IP prefix of the integrated routing and bridging (IRB) interface associated with customer bridge domains or Virtual Extensible LANs (VXLAN). EVPN pure type-5 routing is used when a customer Layer 2 domain is not extended across data centers and the IP subnet for the domain is confined within a single data center. If a BD or VXLAN is not extended to the EVPN, then no EVPN type-5 advertisement for the IP prefix associated with that domain or VXLAN occurs.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373
- EVPN Type-5 Route with MPLS encapsulation for EVPN-MPLS on page 14
- EVPN Type-5 Route with VXLAN encapsulation for EVPN-VXLAN on page 13

bgp-peer

Syntax bgp-peer *ip-address*;

Hierarchy Level [edit routing-instances *name* protocols evpn mclag]

Release Information Statement introduced in Junos OS Release 17.4R1 on MX Series routers, EX Series switches, and Junos Fusion Enterprise.

Description Configure an aggregation device in a Junos Fusion Enterprise or a multichassis link aggregation group (MC-LAG) topology to interwork with an Ethernet VPN-MPLS (EVPN-MPLS) device.

Options ***ip-address***—IP address of the BGP peer. Typically, a BGP peer is identified by the IP address of the device's loopback interface.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG on page 907

designated-forwarder-election-hold-time (evpn)

Syntax

```
designated-forwarder-election-hold-time seconds {
  encapsulation-type
  extended-vni-list
  extended-vni-all
  no-default-gateway-ext-comm
}
```

Hierarchy Level

```
[edit routing-instances protocols evpn]
[edit protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 14.1X53-D15 for QFX Series switches.
Statement introduced in Junos OS Release 18.2R1 for EX Series switches.

Description

A designated forwarder (DF) is required when customer edge devices (CEs) are multihomed to more than one provider edge (PE) device. Without a designated forwarder, multihomed hosts would receive duplicate packets. Designated forwarders are chosen for an Ethernet segment identifier (ESI) based on type 4 route advertisements.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- EVPN Multihoming Overview on page 29

duplicate-mac-detection

Syntax

```
duplicate-mac-detection {
  auto-recovery-time minutes;
  detection-threshold detection-threshold;
  detection-window seconds;
}
```

Hierarchy Level

```
[edit logical-systems  logical-systems-name protocols evpn],
[edit logical-systems logical-systems-name routing-instances  routing-instance-name
  protocols evpn],
[edit protocols evpn]
[edit routing-instances  routing-instance-name protocols evpn],
```

Release Information Statement introduced in Junos OS Release 17.4R1 on MX Series routers, EX Series switches, and QFX Series switches.

Description Configure duplicate MAC address detection settings. You can configure a threshold for MAC address mobility events and the window for detecting the number of MAC address mobility events. In addition, you can configure the optional recovery time that the Juniper Networks device waits before the duplicate MAC address is unsuppressed.

Options

auto-recovery-time—The length of time a device suppresses a duplicate MAC address. At the end of this interval, MAC address updates will resume. When an auto recovery time is not specified, duplicate MAC address updates will continue to be suppressed. To manually clear the suppression of duplicate MAC address, use the **clear evpn duplicate-mac-suppression** command.

Range: 5-360 minutes

detection-threshold—Number of MAC mobility events detected for a MAC address before it is identified as a duplicate MAC address. Once the threshold is reached, updates for this MAC address are suppressed.

Default: 5

Range: 2-20

detection-window—The time interval used in detecting a duplicate MAC address. When the number of MAC mobility events for a MAC address exceeds the **detection-threshold** within the detection window, the MAC address is identified as a duplicate MAC address.

Default: 180 seconds

Range: 5-600 seconds

NOTE: To ensure that the mobility advertisements have sufficient time to age out, set an auto-recovery-time greater than the detection window.

Required Privilege Level routing

Related Documentation

- Overview of MAC Mobility on page 8
- Changing Duplicate MAC Address Detection Settings on page 11

esi

Syntax

```
esi identifier {
  all-active | single-active;
  source-bmac <mac-address>;
}
```

Hierarchy Level

```
[edit interfaces interface-name]
[edit interfaces interface-name unit logical-unit-number]
```

Release Information

Statement introduced in Junos OS Release 14.1.
Statement introduced in Junos OS Releases 15.1F6 and 17.1 on MX Series routers with MPCs.
source-bmac option introduced in Junos OS Release 18.1R1 on MX Series routers with MPCs and MICs.

Description

Configure an Ethernet Segment Identifier (ESI) on a physical, aggregated Ethernet, or logical interface in either EVPN multihoming active-standby or active-active mode.

In releases before Junos OS Release 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI only on a physical or aggregated Ethernet interface, for example, **set interfaces ae0 esi 00:11:22:33:44:55:66:77:88:99**. If you specify an ESI on a physical or aggregated Ethernet interface, keep in mind that an ESI is a factor in the designated forwarder (DF) election process. For example, assume that you configure EVPN multihoming active-standby on aggregated Ethernet interface ae0, and given the ESI configured on ae0 and other determining factors, the DF election results in ae0 being in the down state. Further, all logical interfaces configured on aggregated Ethernet interface ae0, for example, **set interfaces ae0 unit 1** and **set interfaces ae0 unit 2** are also in the down state, which renders logical interfaces ae0.1 and ae0.2 unable to provide services to their respective customer sites (VLANs).

Starting with Junos OS Releases 15.1F6 and 17.1R1 for MX Series routers and Junos OS Release 17.3R1 for EX9200 switches, you can specify an ESI on a logical interface. If you specify an ESI on a logical interface, the DF election process now occurs at the individual logical interface level, which enables you to better utilize logical interfaces. For example, assume that you configure logical interfaces ae0.1 and ae0.2 on aggregated Ethernet interface ae0. You configure EVPN multihoming active-standby on both logical interfaces and given the ESI configured on ae0.1 and other determining factors, the DF election results in ae0.1 being in the down state. Despite logical interface ae0.1 being down, logical interface ae0.2 and other logical interfaces configured on ae0 can be in the up state and provide services to their respective customer sites (VLANs).

Options

esi—Ten octet value. ESI value 0 and all fixed filters are reserved, and not used for configuring a multihomed Ethernet segment.

NOTE:

- Two interfaces (physical, logical, or aggregated Ethernet) cannot be configured with the same ESI value.
- The left most octet must be configured as 00. The other 9 octets are fully configurable.

all-active—Configure the EVPN active-active multihoming mode of operation.

single-active—Configure the EVPN active-standby multihoming mode of operation.

The remaining statements are explained separately. Search for a statement in CLI Explorer or click a linked statement in the Syntax section for details.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- evpn on page 1055
- *source-bmac*
- Example: Configuring an ESI on a Logical Interface With EVPN Multihoming on page 214

evpn

Syntax

```
evpn {
  designated-forwarder-election-hold-timeseconds;
  designated-forwarder-preference-least;
  extended-vni-list {
    vni-options
  }
  no-default-gateway-ext-comm
  encapsulation encapslation-type;
  extended-vni-list;
  multicast-mode client | ingress-replication;
    vni-options{
      vni xxx vrf-target export target:xxx:xx
      vni xxx vrf-export name
    }
  extended-vni-all;
  no-default-gateway-ext-comm;
}
extended-vlan-list vlan-id | [vlan-id set];
extended-isid-list (single-isid | isid-list |isid-range | all);
pbb-evpn-core;
interface interface-name{
  ignore-encapsulation-mismatch;
  interface-mac-limit limit {
    packet-action drop;
  }
  mac-pinning
  no-mac-learning;
  static-mac mac-address;
}
interface-mac-limit limit {
  packet-action drop;
}
label-allocation per-instance;
mac-statistics;
mac-table-size limit {
  packet-action drop;
}
no-mac-learning;
p2mp-bud-support;
  traceoptions {
  file filename <files number> <size size> <world-readable | no-world-readable>;
  flag flag <flag-modifier>;
}
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit protocols]
[edit routing-instances routing-instance-name protocols]
```

Release Information Statement introduced in Junos OS Release 13.2 for EVPNs on MX 3D Series routers.
designated-forwarder-election-hold-time *seconds* statement introduced in Junos OS Release 14.1.
extended-vlan-list *vlan-id* | [*vlan-id set*] statement introduced in Junos OS Release 14.1. Statement introduced in Junos OS Release 14.1-X53-D30 for QFX Series switches.

NOTE: The **extended-vlan-list** statement is not supported on QFX10000 switches.

designated-forwarder-preference-least and **service-type** statements introduced in Junos OS Release 17.3 for MX Series routers with MPC and MIC interfaces.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.
p2mp-bud-support statement introduced in Junos OS Release 18.2R1.

Description Enable an Ethernet VPN (EVPN) on the routing instance.

Options

designated-forwarder-election-hold-time *seconds*—Time in seconds to wait before electing a designated forwarder (DF).
Range: 1 through 1800 seconds

designated-forwarder-preference-least—Use least preference value for DF election. By default, the preference-based DF election is based on the highest preference value configured.

evpn-etree—Configure an Ethernet VPN E-TREE service.

p2mp-bud-support—Enables bud node support on the device. That is, this device can function as both a transit and egress PE.

NOTE: Enabling or disabling **p2mp-bud-support** may result in a temporary loss of packets. This occurs because changing bud node support affects the forwarding state in the routing instance, which causes the forwarding table to be rebuilt.

The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Implementing EVPN-VXLAN for Data Centers on page 272
- Configuring EVPN Routing Instances on page 3
- Tracing EVPN Traffic and Operations on page 19
- Implementing EVPN-VXLAN for Data Centers on page 272
- Provider Backbone Bridging (PBB) and EVPN Integration Overview on page 953

extended-isid-list

Syntax

```
extended-isid-list (single-isid | isid-list |isid-range | all);
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit routing-instances routing-instance-name protocol evpn]
```

Release Information

Statement introduced in Junos OS Release 16.1.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

Specify the service identifiers (ISIDs) that are being extended over the Ethernet VPN (EVPN) core.

NOTE: When you deactivate the extended-isid-list statement, with or without configuring the automatic derivation of the BGP route target (auto-RT) for advertised prefixes, both Type 2 and Type 3 routes are retained.

Options

single-isid—Specify a single ISID.

isid-list—Specify a list of multiple ISIDs.

isid-range—Specify a range of ISIDs.

all—Specify that all ISIDs are being extended over the EVPN core.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- evpn on page 1055

extended-vlan-list

Syntax extended-vlan-list *vlan-id* | [*vlan-id set*];

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols]
[edit routing-instances *routing-instance-name* instance-type virtual-switch protocols evpn]

Release Information Statement introduced in Junos OS Release 14.1.
Statement introduced in Junos OS Release 14.2 on EX Series switches.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description Specify the VLAN or range of VLANs that are extended over the WAN, wherein all the single VLAN bridge domains corresponding to these VLANs are stretched.

NOTE: The extended-vni-list statement is an exclusive command. You cannot configure the extended-vni-list statement with either the extended-vlan-list or extended-vni-all statements.

Options ***vlan-id***—VLAN ID to be EVPN extended.

vlan-id set—List of VLAN IDs to be EVPN extended.

Range: 1 through 4094 VLANs

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- evpn on page 1055

global-mac-ip-limit

Syntax

```
global-mac-ip-limit {
  number;
}
```

Hierarchy Level

```
[edit logical-systems name protocols l2-learning],
[edit protocols l2-learning]
```

Release Information

Statement introduced in Junos OS Release 17.4R2 for MX Series routers and EX Series switches.
Statement introduced in Junos OS Release 18.1R1 for QFX Series switches.

Description

Limit the number of entries that can be added systemwide to the MAC-IP bindings database.

Options

number—Specify the maximum number of entries that can added systemwide to the MAC-IP bindings database.. When the specified maximum is reached, no new entries are added to the database.

Range: 20 through 1,048,575.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression on page 227
- global-mac-ip-table-aging-time on page 1060
- interface-mac-ip-limit on page 1065
- mac-ip-table-size on page 1074

global-mac-ip-table-aging-time

Syntax

```
global-mac-ip-table-aging-time seconds;
```

Hierarchy Level

```
[edit logical-systems logical-system-name protocols l2-learning],
[edit protocols l2-learning]
```

Release Information Statement introduced in Junos OS Release 17.4R1 for MX Series routers and EX9200 switches.

Description Configure the timeout interval systemwide for entries in the MAC-IP address bindings database.

Options

seconds—Specify the time that is elapsed before entries in the MAC-IP bindings database are timed out and deleted.

Range: For MX Series routers, 10 through 1 million; for EX9200 switches, 60 through 1 million.

Default: 1200 (20 minutes)

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression on page 227
- global-mac-ip-limit on page 1059
- interface-mac-ip-limit on page 1065
- mac-ip-table-size on page 1074

instance-type

Syntax instance-type *type*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
virtual-switch and **layer2-control** options introduced in Junos OS Release 8.4.
Statement introduced in Junos OS Release 9.2 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.
mpls-internet-multicast option introduced in Junos OS Release 11.1 for the EX Series, M Series, MX Series, and T Series.
evpn option introduced in Junos OS Release 13.2 for MX 3D Series routers.
evpn option introduced in Junos OS Release 17.3 for the QFX Series.
forwarding option introduced in Junos OS Release 14.2 for the PTX Series.
mpls-forwarding option introduced in Junos OS Release 16.1 for the MX Series.
evpn-vpws option introduced in Junos OS Release 17.1 for MX Series routers.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description Define the type of routing instance.

Options

NOTE: On ACX Series routers, you can configure only the forwarding, virtual router, and VRF routing instances.

type—Can be one of the following:

- **evpn**—(MX 3D Series routers, QFX switches and EX9200 switches)—Enable an Ethernet VPN (EVPN) on the routing instance.
  hierarchy level.
- **evpn-vpws**—Enable an Ethernet VPN (EVPN) Virtual Private Wire Service (VPWS) on the routing instance.
- **forwarding**—Provide support for filter-based forwarding, where interfaces are not associated with instances. All interfaces belong to the default instance. Other instances are used for populating RPD learned routes. For this instance type, there is no one-to-one mapping between an interface and a routing instance. All interfaces belong to the default instance inet.0.
- **l2backhaul-vpn**—Provide support for Layer 2 wholesale VLAN packets with no existing corresponding logical interface. When using this instance, the router learns both the outer tag and inner tag of the incoming packets, when the **instance-role** statement is

defined as **access**, or the outer VLAN tag only, when the **instance-role** statement is defined as **nni**.

- **l2vpn**—Enable a Layer 2 VPN on the routing instance. You must configure the **interface**, **route-distinguisher**, **vrf-import**, and **vrf-export** statements for this type of routing instance.
- **layer2-control**—(MX Series routers only) Provide support for RSTP or MSTP in customer edge interfaces of a VPLS routing instance. This instance type cannot be used if the customer edge interface is multihomed to two provider edge interfaces. If the customer edge interface is multihomed to two provider edge interfaces, use the default BPDU tunneling.
- **mpls-forwarding**—(MX Series routers only) Allow filtering and translation of route distinguisher (RD) values in IPv4 and IPv6 VPN address families on both routes received and routes sent for selected BGP sessions. In particular, for Inter-AS VPN Option-B networks, this option can prevent the malicious injection of VPN labels from one peer AS boundary router to another.
- **mpls-internet-multicast**—(EX Series, M Series, MX Series, and T Series routers only) Provide support for ingress replication provider tunnels to carry IP multicast data between routers through an MPLS cloud, using MBGP or next-generation MVPN.
- **no-forwarding**—This is the default routing instance. Do not create a corresponding forwarding instance. Use this routing instance type when a separation of routing table information is required. There is no corresponding forwarding table. All routes are installed into the default forwarding table. IS-IS instances are strictly nonforwarding instance types.
- **virtual-router**—Enable a virtual router routing instance. This instance type is similar to a VPN routing and forwarding instance type, but used for non-VPN-related applications. You must configure the **interface** statement for this type of routing instance. You do not need to configure the **route-distinguisher**, **vrf-import**, and **vrf-export** statements.
- **virtual-switch**—(MX Series routers, EX9200 switches, and QFX switches only) Provide support for Layer 2 bridging. Use this routing instance type to isolate a LAN segment with its Spanning Tree Protocol (STP) instance and to separate its VLAN identifier space.
- **vpls**—Enable VPLS on the routing instance. Use this routing instance type for point-to-multipoint LAN implementations between a set of sites in a VPN. You must configure the **interface**, **route-distinguisher**, **vrf-import**, and **vrf-export** statements for this type of routing instance.
- **vrf**—VPN routing and forwarding (VRF) instance. Provides support for Layer 3 VPNs, where interface routes for each instance go into the corresponding forwarding table only. Required to create a Layer 3 VPN. Create a VRF table (***instance-name*.inet.0**) that contains the routes originating from and destined for a particular Layer 3 VPN. For this instance type, there is a one-to-one mapping between an interface and a routing instance. Each VRF instance corresponds with a forwarding table. Routes on an interface go into the corresponding forwarding table. You must configure the **interface**, **route-distinguisher**, **vrf-import**, and **vrf-export** statements for this type of routing instance.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring the Instance Type*
- Configuring EVPN Routing Instances on page 3
- Configuring EVPN Routing Instances on EX9200 Switches on page 6
- *Configuring Virtual Router Routing Instances*
- *Example: Configuring Filter-Based Forwarding on the Source Address*
- *Example: Configuring Filter-Based Forwarding on Logical Systems*

interface (EVPN Routing Instances)

Syntax

```
interface interface-name {
  ignore-encapsulation-mismatch;
  interface-mac-limit limit {
    packet-action drop;
  }
  mac-pinning
  no-mac-learning;
  protect-interface
  static-mac mac-address;
  vpws-service-id
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 13.2 for EVPNs on MX 3D Series routers.
Statement introduced in Junos OS Release 14.2 on EX Series switches.
Statement (mac-pinning) introduced in Junos OS Release 16.2 on MX Series routers.
vpws-service-id statement introduced in Junos OS Release 17.1 on MX Series routers.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

Specify each interface over which the Ethernet VPN (EVPN) traffic travels between the PE device and customer edge (CE) device. The interfaces are bound to the EVPN routing instance.

Options

interface-name—Name of the interface.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 3
- Configuring EVPN Routing Instances on EX9200 Switches on page 6
- evpn on page 1055
- instance-type on page 1061
- vpws-service-id on page 1090

interface-mac-ip-limit

Syntax

```
interface-mac-ip-limit number;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name vlans
  vlan-name switch-options],
[edit logical-systems logical-system-name vlans vlan-name switch-options],
[edit routing-instances routing-instance-name protocols evpn],
[edit routing-instances routing-instance-name bridge-domains bridge-domain-name
  bridge-options],
[edit routing-instances routing-instance-name vlans vlan-name switch-options],
[edit vlans vlan-name switch-options]
```

Release Information

Statement introduced in Junos OS Release 17.4R2 on MX Series routers and EX Series switches.
Statement introduced in Junos OS Release 18.1R1 on QFX Series switches.

Description

Limit the number of entries that can be learned on an interface through the MAC-IP bindings database. Depending on the Juniper Networks device, this limit can be applied to EVPN instances, bridge domains configured in a virtual-switch routing instance, or VLANs configured in a virtual-switch routing instance.

To apply this limit systemwide, use the global-mac-ip-limit statement at the **[edit protocols l2-learning]** hierarchy level.

Options

number—Specify the maximum number of MAC-IP bindings per interface. After that maximum is reached, no additional MAC-IP entries are added to the database.

Range: 1 through 1024.

Default: 124

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression on page 227
- mac-ip-table-size on page 1074

interface-mac-limit (VPLS)

Syntax

```
interface-mac-limit limit {
  packet-action drop;
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
  vpls site site-name interfaces interface-name],
[edit routing-instances routing-instance-name protocols evpn],
[edit routing-instances routing-instance-name protocols evpn interface interface-name],
[edit routing-instances routing-instance-name protocols vpls site site-name interfaces
  interface-name]
```

Release Information

Statement introduced before Junos OS Release 7.4.
Support for EVPNs introduced in Junos OS Release 13.2 on MX 3D Series routers.
Support for EVPNs introduced in Junos OS Release 14.2 on EX Series switches.

Description

Specify the maximum number of media access control (MAC) addresses that can be learned by the EVPN or VPLS routing instance. You can configure the same limit for all interfaces configured for a routing instance. You can also configure a limit for a specific interface.

Starting with Junos OS Release 12.3R4, if you do not configure the parameter to limit the number of MAC addresses to be learned by a VPLS instance, the default value is not effective. Instead, if you do not include the **interface-mac-limit** option at the **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name* interfaces *interface-name*],** hierarchy level, this setting is not present in the configuration with the default value of 1024 addresses. If you upgrade a router running a Junos OS release earlier than Release 12.3R4 to Release 12.3R4 or later, you must configure the **interface-mac-limit** option with a valid value for it to be saved in the configuration.

Options

limit—Number of MAC addresses that can be learned from each interface.

Range: 1 through 131,071 MAC addresses

NOTE: For M120 devices only, the range is 16 through 65,536 MAC addresses.

Default: 1024 addresses

The remaining statement is explained separately. Search for a statement in CLI Explorer or click a linked statement in the Syntax section for details.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 3
- Configuring EVPN Routing Instances on EX9200 Switches on page 6
- *Limiting the Number of MAC Addresses Learned from an Interface*
- interface on page 1064
- *mac-table-size*

ip-prefix-routes

Syntax

```
ip-prefix-routes {
  advertise (direct-nexthop | gateway-address);
  encapsulation (vxlan | mpls);
  <export routing-policy-name>;
  gateway-interface interface-name;
  vni number;
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
  evpn]
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D60 and Junos OS Release 17.2R1 for QFX10000 switches.
Statement introduced in Junos OS Release 17.3R1 for EX9200 switches.
Statement introduced in Junos OS Release 17.4R1 for QFX5110 switches.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

In an Ethernet VPN (EVPN) environment, enable the switch to advertise the IP prefix associated with a specified customer domain as a type-5 route to remote data centers or in a metro transport network. You use this feature when the Layer 2 domain does not exist at the remote data centers or metro network peering points.

Two models for implementing type-5 routes are supported:

- Pure type-5 route without overlay next hop and type-2 route
- Type-5 route with gateway IRB interface as overlay next hop and type-2 route

A pure type-5 route advertises the summary IP prefix and includes a BGP extended community called a router MAC, which is used to carry the MAC address of the sending switch and to provide next-hop reachability information for the prefix. In contrast, a standard type-5 route requires a gateway IP address as a next-hop overlay and a supporting type-2 route to provide recursive route resolution.

On QFX5110 and QFX10000 switches, only fully resolved next-hop—that is, EVPN pure type-5—routes are currently supported. QFX10000 switches support only EVPN Virtual Extensible LAN (VXLAN). MPLS encapsulation is not supported.

On QFX5110 switches, you must also configure the overlay-ecmp statement at the **[edit forwarding-options vxlan-routing]** hierarchy level with pure type-5 routes in an overlay EVPN-VXLAN network. We strongly recommend also configuring the overlay-ecmp statement whenever you enable pure type-5 routes on a QFX5110 switch. Configuring this statement causes the Packet Forwarding Engine to restart. Restarting the Packet Forwarding Engine interrupts all forwarding operations. Therefore, we strongly recommend configuring the **overlay-emcp** statement before the EVPN-VXLAN network becomes operational.

On EX9200 switches, both type-5 routes are supported. Only MPLS encapsulation is supported with pure type-5 routes. Both MPLS and VXLAN encapsulation are supported with the standard type-5 route with a gateway IP address as next-hop overlay.

CAUTION: Pure type-5 routing for EVPN-VXLAN was introduced in Junos OS Release 15.1X53-D30 for QFX10002 switches only. In that release, this statement is **ip-prefix-support forwarding-mode symmetric**.
Starting with Junos OS Release 15.1X53-D60, the statement is **ip-prefix-routes advertise direct-nexthop**.
Any configuration with the original **ip-prefix-support** statement is automatically upgraded to the new **ip-prefix-routes** statement when you upgrade to Junos OS Release 15.1X53-D60 or later.

NOTE: Pure type-5 routing is supported on all QFX10000 switches starting in Junos OS Release 15.1X53-D60.

NOTE: QFX10000 switches do not support advertising an IP prefix with a mask length of /32 as a pure type-5 route.

Options The **advertise**, **encapsulation** and **vni** ***number*** options are required. The **export** ***routing–policy-name*** option is optional.

advertise direct-nexthop—Enable the switch to send IP prefix information using an EVPN pure type-5 route, which includes a router MAC extended community used to send the MAC address of the switch. This router MAC extended community provides next-hop reachability without requiring an overlay next-hop or supporting type-2 route.

NOTE: For pure route type-5, QFX5110 and QFX10000 switches support only VXLAN encapsulation, and EX9200 switches support only MPLS encapsulation.

advertise gateway-address (EX9200 switches only)—Enable the switch to advertise a gateway address in exported IP prefix routes. This gateway address provides overlay next-hop reachability.

NOTE: You must also specify a gateway address by including the **gateway-interface** ***interface-name*** statement.

encapsulation (vxlan | mpls)—Specify to encapsulate forwarded traffic in VXLAN or MPLS for transmission to the remote data center.

NOTE: The same type of encapsulation must be used end to end. Only VXLAN encapsulation is supported on QFX10000 and QFX5110 switches.

export ***routing-policy-name***—(Optional) Specify the name of the routing policy configured at the **[edit policy-options policy-statement** ***policy-statement-name*****]** hierarchy level to apply to the routes for the specified customer domain. Applying an export policy allows you to further control the IP prefixes to advertise or to suppress through EVPN type-5 routes for each customer. You can apply a separate export routing policy to one or more customer domains. This allows each customer to each have its own policy.

gateway-interface ***interface-name***—Specify the gateway interface to use as a next-hop overlay for a standard type-5 route. You must use this option in conjunction with the **advertise gateway-address** option.

vni ***number***—Specify the identifier associated with a customer domain. Each customer domain must have a unique identifier.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN Pure Type-5 Routes on page 15
- EVPN Overview for Switches on page 552
- *policy-statement*

ip-prefix-support

Syntax

```
ip-prefix-support {
  encapsulation vxlan;
  <export routing-policy-name>;
  forwarding-mode symmetric;
  vni number;
}
```

Hierarchy Level

[edit routing-instances *routing-instance-name* protocols evpn]

Release Information

Statement introduced in Junos OS Release 15.1x53-D30 on QFX10002 switches.

Description

In an Ethernet VPN (EVPN) Virtual Extensible LAN (VXLAN) environment, enable the provider edge (PE) switch to forward the IP prefix associated with a specified customer domain in scenarios where the Layer 2 domain does not extend across data centers. Such routes are referred to as EVPN pure type-5 routes. On QFX switches, only fully resolved next-hop routes are supported. The data packets are sent as Layer 2 frames that are encapsulated in the VXLAN header. A unique VXLAN numerical identifier is associated with each customer domain.

When the advertisement of the type-5 route is enabled, you can optionally use an export routing policy configured at the **[edit policy-options policy-statement *policy-name*]** hierarchy level to further control the IP prefixes to advertise or suppress through EVPN type-5 routes for each customer. The policy is applied to the routes that reside in the customer's inet.0 routing table by specifying a ***routing-instance-name*** configured at the **[edit routing-instances]** hierarchy level. This ***routing-instance-name*** refers to a Layer 3 virtual routing and forwarding (VRF) domain for a specific customer. The corresponding name of the table is ***instance.name*.inet.0** This table is created by default when you configure a routing instance.

CAUTION: This statement is deprecated starting with Junos OS Release 15.1x53-D60 and has been replaced with ip-prefix-routes. The statement **ip-prefix-support forwarding-mode symmetric** is now **ip-prefix-routes advertise direct-nexthop**. Any configuration with the original **ip-prefix-support** statement is automatically upgraded to the new **ip-prefix-routes** statement when you upgrade to Junos OS Release 15.1x53-D60 or later.

Options

The **forwarding-mode symmetric**, **encapsulation vxlan** and **vni *number*** options are required. The **export *policy-name*** option is optional.

encapsulation vxlan—Specify to encapsulate forwarded traffic in VXLAN for transmission to the other data center.

export *routing-policy-name*—(Optional) Specify the name of the routing policy configured at the **[edit policy-options policy-statement *policy-name*]** hierarchy level to apply to the routes for the specified customer domain. Applying an export policy allows you to further control the IP prefixes to advertise or to suppress for each customer. You can apply a separate export routing policy to one or more customer domains. This allows each customer to each have its own policy.

forwarding-mode symmetric—Enable the receiver to resolve the next-hop route using only information carried in the type-5 route.

vni *number*—Specify the identifier associated with a customer domain. Each customer domain must have a unique identifier.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN Pure Type-5 Routes on page 15
- *policy-statement*

mac-ip-table-size

Syntax

```
mac-ip-table-size number;
```

Hierarchy Level

```
[edit logical-systems name routing-instances routng-instance-name protocols evpn],
[edit routing-instances routing-instance-name protocols evpn],
[edit routing-instances routing-instance-name bridge-domains bridge-domain-name
  bridge-options],
[edit routing-instances routing-instance-name vlans vlan-name switch-options]
```

Release Information

Statement introduced in Junos OS Release 17.4R2 for MX Series routers and EX Series switches.
Statement introduced in Junos OS Release 18.1R1 for QFX Series switches.

Description

Limit the number of entries that can be added to the MAC-IP address bindings database. Depending on the Juniper Networks device, this limit can be applied to EVPN routing instances, bridge domains configured in a virtual-switch routing instance, or VLANs configured in a virtual-switch routing instance. When the specified maximum number of entries is reached, no additional entries are added to the MAC-IP bindings database.

To apply this limit systemwide, use the global-mac-ip-limit statement at the **[edit protocols l2-learning]** hierarchy level.

Options

number—Maximum number of entries that can be added to the MAC-IP address bindings database.

Range: 16 through 1,048,575.

Default: 8192

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression on page 227
- interface-mac-ip-limit on page 1065

mac-pinning (EVPN Routing Instances)

Syntax

```
mac-pinning;
```

Hierarchy Level

```
[edit routing-instances routing-instance-name protocols protocol interface interface-name]
[edit logical-systems logical-system-name routing-instances routing-instance-name vlans
    vlan-namebridge-domains bridge-name bridge-options-interface interface-name]
[edit logical-systems logical-system-name routing-instances routing-instance-name vlans
    vlan-namerouting-instances routing-instance-name switch-options interface
    interface-name]
```

Release Information Statement introduced in Junos OS Release 16.2 for EVPNs.

Description Sets mac-pinning on the interface, which makes it static. That is, the MAC-pinned address cannot be moved to any other interface in the bridge domain.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- EVPN MAC Pinning Overview on page 232
- Configuring EVPN MAC Pinning on page 234
- *Configuring MAC Pinning for PBB-EVPN*
- *Understanding Layer 2 Learning and Forwarding for Bridge Domains*
- *Understanding Layer 2 Learning and Forwarding for Bridge Domains Functioning as Switches with Layer 2 Trunk Ports*

mclag

Syntax

```
mclag {
  bgp-peer ip-address;
}
```

Hierarchy Level

[edit routing-instances *name* protocols evpn]

Release Information

Statement introduced in Junos OS Release 17.4R1 on MX Series routers, EX Series switches, and Junos Fusion Enterprise.

Description

Configure parameters that enable the interworking of Ethernet VPN-MPLS (EVPN-MPLS) with a Junos Fusion Enterprise or a multichassis link aggregation group (MC-LAG) topology.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN-MPLS Interworking with Junos Fusion Enterprise and MC-LAG on page 907

multicast-mode (EVPN)

Syntax

```
multicast-mode client | ingress-replication;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name
  protocols],
[edit protocols evpn],
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.
Statement introduced in Junos OS Release 17.3R1 for EX Series switches.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

Configure the multicast server mode for delivering traffic and packets for Ethernet VPN (EVPN). This statement is required for a VXLAN EVPN instance.

NOTE: If you configure the **multicast-mode** statement, then you must also configure the **encapsulation vxlan** statement.

Options

client—Use the client as the multicast mode for delivering traffic and multicast packets across routers and switches.

ingress-replication—Use ingress replication as the multicast mode for delivering broadcast, unknown unicast, and multicast (BUM) traffic and multicast packets across routers and switches.

Default: ingress-replication

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches on page 270
- Using a Default Layer 3 Gateway to Route Traffic Between Virtual Networks in an EVPN-VXLAN Topology on page 293
- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373

no-arp-suppression

Syntax

```
no-arp-suppression;
```

Hierarchy Level

The instance type 'EVPN' supports under the hierarchy:

[edit bridge-domains *bridge-domain-name*]
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols]
[edit routing-instances *instance-name* protocols evpn]
[edit routing-instances *instance-name* bridge-domains *domain-name*]

The instance type 'virtual-switch' supports under hierarchy:

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* vlans *vlan-name*]
[edit routing-instances *routing-instance-name* vlans *vlan-name*]
[edit vlans *vlan-name*]

Release Information

Statement introduced in Junos OS Release 17.2R1 for MX Series routers and EX Series switches.
Statement introduced in Junos OS Release 17.3R1 for QFX Series switches.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

Disable the suppression of Address Resolution Protocol (ARP) requests from a customer edge (CE) device or Layer 2 VXLAN gateway to a provider edge (PE) device or Layer 3 VXLAN gateway that acts as ARP proxy in an Ethernet VPN-MPLS (EVPN-MPLS) or Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) environment.

When disabling ARP suppression, be aware of the following implications;

- If the PE device or Layer 3 VXLAN gateway does not find the MAC-IP address binding in its database, it cannot forward the ARP request. Instead, the device discards the ARP request, and the packets flood through the Layer 2 domain because the ARP request is considered to be Layer 2 broadcast traffic.
- Disabling the suppression of ARP packets on a PE device or Layer 3 VXLAN gateway essentially disables the proxy ARP functionality.

Therefore, we recommend that ARP suppression remains enabled.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- EVPN Proxy ARP and ARP Suppression, and NDP and NDP Suppression on page 227

no-default-gateway-ext-comm

Syntax

```
no-default-gateway-ext-comm;
```

Hierarchy Level

```
[edit routing-instances name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 16.1.

Description

Configure a routing instance to suppress the advertisement of the extended community on the default gateway. An extended community is an 8-octet community used by Ethernet VPNs (EVPNs).

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Implementing EVPN-VXLAN for Data Centers on page 272
- EVPN Multihoming Overview on page 29

nsr-phantom-holdtime

Syntax nsr-phantom-holdtime *seconds*;

Hierarchy Level [edit routing-options]

Release Information Statement introduced in Junos OS Release 15.1x53-D60 for QFX10000 switches.

Description Specify to hold phantom IP addresses, that is, prevent these routes from being added to routing tables, for a specified period of time. During this hold-time interval, these routes are maintained in the kernel. After the hold time expires, the routes are added to the routing tables.

We strongly recommend that you configure this statement before you perform a graceful Routing Engine switchover (GRES) when nonstop routing (NSR) is enabled. Doing so prevents traffic loss because routes are added to the routing tables after the hold-time interval expires, but before they are deleted from the kernel during a switchover.

Options **seconds**—Specify the interval of time to prevent phantom IP addresses from being added to the routing tables. After this interval expires, the routes are added to the routing tables.

BEST PRACTICE: In an EVPN/VXLAN environment with NSR and GRES enabled, the recommended hold-time value is 300 (5 minutes)

Range: 0 through 10000

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Understanding Graceful Routing Engine Switchover*
- Preventing Traffic Loss in an EVPN/VXLAN Environment With GRES and NSR on page 238
- *Example: Configuring Nonstop Active Routing on Switches*

overlay-ecmp

Syntax overlay-ecmp;

Hierarchy Level [edit forwarding-options vxlan-routing]

Release Information Statement introduced in Junos OS Release 17.4R1 for QFX5110 switches.

Description Enable two-level equal-cost multipath next hops. This statement is required for a Layer 3 Ethernet VPN Virtual Extensible LAN (EVPN-VXLAN) overlay network when pure type-5 routing is also configured. It is also strongly recommend that you configure this statement whenever pure type-5 routes are enabled on QFX5110 switches. For more information about configuring pure type-5 routes, see ip-prefix-routes.

NOTE: Configuring this statement causes the Packet Forwarding Engine to restart. Restarting the Packet Forwarding Engine interrupts all forwarding operations. Therefore, we strongly recommend using this configuration statement before the EVPN-VXLAN network becomes operational.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- vxlan-routing on page 1112

pbb-evpn-core

Syntax

```
pbb-evpn-core;
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit routing-instances routing-instance-name protocol evpn]
```

Release Information

Statement introduced in Junos OS Release 16.1.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

Specify that Ethernet VPN (EVPN) is running for Provider Backbone Bridging (PBB).

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- evpn on page 1055

proxy-macip-advertisement

Syntax proxy-macip-advertisement;

Hierarchy Level [edit interfaces irb unit *logical-unit-number*]

Release Information Statement introduced in Junos OS Release 15.1X53-D60 for QFX Series switches.

Description Enable the proxy advertisement feature on a QFX Series switch that can function as a Layer 3 gateway. With this feature enabled, the Layer 3 gateway advertises the MAC and IP routes (MAC+IP type 2 routes) on behalf of Layer 2 VXLAN gateways.

In an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) topology with a two-layer IP fabric, spine devices typically function as Layer 3 VXLAN gateways, and leaf devices typically function as Layer 2 gateways. In this topology, the Layer 2 VXLAN gateways can advertise only the MAC routes (EVPN type 2 routes) for the attached hosts. Since the Layer 2 gateways are unable to resolve the MAC-to-IP bindings for the hosts, each of the Layer 3 gateways rely on the Address Resolution Protocol (ARP) and the Neighbor Discovery Protocol (NDP) to discover and install the bindings.

For example, after a Layer 3 gateway receives a host MAC route advertisement from a Layer 2 gateway, and ARP and NDP resolve the MAC-to-IP bindings, the Layer 3 gateway in turn advertises the host MAC and IP routes along with the next hop, which is set to the Layer 2 gateway to which the host is attached. Upon receipt of this advertisement, Layer 2 and 3 gateways in the topology install the MAC-to-IP bindings along with the associated next hops. When any of these gateways receives a packet with a destination MAC that matches an address in its MAC table, the gateway can check the next hop associated with the MAC address and forward the packet directly to the Layer 2 gateway to which the host is attached. This resulting packet flow eliminates the need for the packet to be forwarded first to a Layer 3 gateway, which then forwards the packet to the Layer 2 gateway.

Enabling this feature in an EVPN-VXLAN topology with a two-layer fabric is mandatory. Enabling this feature in an EVPN-VXLAN topology where the IP fabric is collapsed into a single layer of QFX Series switches that function as both Layer 2 and Layer 3 VXLAN gateways is optional.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

route-distinguisher

Syntax route-distinguisher (*as-number:id* | *ip-address:id*);

Hierarchy Level
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn mesh-group *mesh-group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*],
[edit routing-instances *routing-instance-name*],
[edit routing-instances *routing-instance-name* protocols l2vpn mesh-group *mesh-group-name*],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.
Support at **[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]** hierarchy level introduced in Junos OS Release 11.2.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.
Support at **[edit routing-instances *routing-instance-name* protocols l2vpn mesh-group *mesh-group-name*]** hierarchy level introduced in Junos OS Release 13.2.
Statement introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.

Description
Specify an identifier attached to a route, enabling you to distinguish to which VPN or virtual private LAN service (VPLS) the route belongs. Each routing instance must have a unique route distinguisher (RD) associated with it. The RD is used to place bounds around a VPN so that the same IP address prefixes can be used in different VPNs without having them overlap. If the instance type is **vrf**, the **route-distinguisher** statement is required.

For Layer 2 VPNs and VPLS, if you configure the **l2vpn-use-bgp-rules** statement, you must configure a unique RD for each PE router participating in the routing instance.

For other types of VPNs, we recommend that you use a unique RD for each provider edge (PE) router participating in specific routing instance. Although you can use the same RD on all PE routers for the same VPN routing instance, if you use a unique RD, you can determine the customer edge (CE) router from which a route originated within the VPN.

For Layer 2 VPNs and VPLSs, if you configure mesh groups, the RD in each mesh group must be unique.

CAUTION: We strongly recommend that if you change an RD that has already been configured, make the change during a maintenance window, as follows:

1. Deactivate the routing instance.
2. Change the RD.

3. Activate the routing instance.

This is not required if you are configuring the RD for the first time.

Options

as-number:number*—*as-number is an assigned AS number, and ***number*** is any 2-byte or 4-byte value. The AS number can be from 1 through 4,294,967,295. If the AS number is a 2-byte value, the administrative number is a 4-byte value. If the AS number is 4-byte value, the administrative number is a 2-byte value. An RD consisting of a 4-byte AS number and a 2-byte administrative number is defined as a type 2 RD in RFC 4364 *BGP/MPLS IP VPNs*.

NOTE: In Junos OS Release 9.1 and later, the numeric range for AS numbers is extended to provide BGP support for 4-byte AS numbers, as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*. All releases of Junos OS support 2-byte AS numbers. To configure an RD that includes a 4-byte AS number, append the letter "L" to the end of the AS number. For example, an RD with the 4-byte AS number 7,765,000 and an administrative number of 1,000 is represented as 77765000L:1000.

In Junos OS Release 9.2 and later, you can also configure a 4-byte AS number using the AS dot notation format of two integer values joined by a period: *<16-bit high-order value in decimal>.<16-bit low-order value in decimal>*. For example, the 4-byte AS number of 65,546 in the plain-number format is represented as 1.10 in AS dot notation format.

ip-address*:*id—IP address (***ip-address*** is a 4-byte value) within your assigned prefix range and a 2-byte value for the ***id***. The IP address can be any globally unique unicast address.

Range: 0 through 4,294,967,295 ($2^{32} - 1$). If the router you are configuring is a BGP peer of a router that does not support 4-byte AS numbers, you need to configure a local AS number. For more information, see *Using 4-Byte Autonomous System Numbers in BGP Networks Technology Overview*.

NOTE: For Ethernet VPN (EVPN), an RD that includes zero as the *id* value is reserved for the default EVPN routing instance by default. Because the same RD cannot be assigned for two routing instances, using a *ip-address*:*id* RD for another routing instance (default-switch), where the *id* value is zero, throws a commit error.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring BGP Route Target Filtering for VPNs*
- *Example: Configuring FEC 129 BGP Autodiscovery for VPWS*
- Configuring EVPN Routing Instances on page 3
- *Configuring the Route Distinguisher*
- *Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)*
- *Configuring an MPLS-Based Layer 3 VPN (CLI Procedure)*
- *l2vpn-use-bgp-rules*

traceoptions (Protocols EVPN)

Syntax

```
traceoptions {
  file filename <files number> <size size> <world-readable | no-world-readable>;
  flag flag <flag-modifier>;
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 13.2 for MX 3D Series routers.
Statement introduced in Junos OS Release 14.2 for EX Series switches.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

Trace traffic flowing through an EVPN routing instance.

Options

file *filename*—Name of the file to receive the output of the tracing operation. Enclose the name in quotation marks (" ").

files *number*—(Optional) Maximum number of trace files. When a trace file named ***trace-file*** reaches the maximum size as specified by the **size** option, it is renamed ***trace-file*.0**. When ***trace-file*** again reaches the maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

Range: 2 through 1000 files

Default: 2 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements. You can specify the following tracing flags:

- **all**—All EVPN tracing options
- **error**—Error conditions
- **general**—General events
- **mac-database**—MAC route database in the EVPN routing instance
- **nlri**—EVPN advertisements received or sent by means of BGP
- **normal**—Normal events
- **oam**—OAM messages
- **policy**—Policy processing
- **route**—Routing information

- **state**—State transitions
- **task**—Routing protocol task processing
- **timer**—Routing protocol timer processing
- **topology**—EVPN topology changes caused by reconfiguration or advertisements received from other provider edge (PE) routers using BGP

flag-modifier—(Optional) Modifier for the tracing flag. You can specify the following modifiers:

- **detail**—Provide detailed trace information.
- **disable**—Disable this trace flag.
- **receive**—Trace received packets.
- **send**—Trace sent packets.

no-world-readable—Do not allow any user to read the log file.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When ***trace-file*** again reaches the maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files (as specified by the **files** option) is reached. Then the oldest trace file is overwritten.

If you specify a maximum file size, you also must specify a maximum number of trace files with the **files** option.

Syntax: ***x*k** to specify kilobytes, ***x*m** to specify megabytes, or ***x*g** to specify gigabytes

Range: 10 KB through the maximum file size supported on your system

Default: 1 MB

world-readable—Allow any user to read the log file.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Tracing EVPN Traffic and Operations on page 19

vlan-id (routing instance)

Syntax

```
vlan-id (vlan-id | all | none);
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit routing-instances routing-instance-name]
[edit routing-instances routing-instance-name instance-type]
```

Release Information

Statement introduced in Junos OS Release 13.2.
Statement introduced in Junos OS Release 14.2 for EX Series switches.
Statement introduced in Junos OS Release 17.1 for the QFX Series.

Description

Specify 802.1Q VLAN tag IDs to a routing instance.

Options

vlan-id—A valid VLAN identifier.

Range: For 4-port Fast Ethernet PICs, 512 through 1023. For 1-port and 10-port Gigabit Ethernet PICs configured to handle VPLS traffic, 512 through 4094.

all—Include all VLAN identifiers specified on the logical interfaces included in the routing instance.

none—Include no VLAN identifiers for the routing instance.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring EVPN Routing Instances on page 3
- Configuring EVPN Routing Instances on EX9200 Switches on page 6

vpws-service-id

Syntax

```
vpws-service-id {
  local service-id;
  remote serivce-id;
}
```

Hierarchy Level

[edit routing-instance *instance-type* protocols evpn interface *interface-name*]

Release Information

Statement introduced in Junos OS Release 17.1 for MX Series routers.

Description

Specify the local and remote Ethernet VPN (EVPN)-Virtual Private Wire Service (VPWS) service identifiers. These service identifiers are unique to an EVPN and are used to identify the endpoints of the EVPN-VPWS network. These endpoints are autodiscovered by BGP and are used to exchange the service labels (learned from the respective provider edge (PE) routers) that are used by autodiscovered routes per EVI route type. Depending on the mode of operation of the PE routers in the EVPN-VPWS network, these two endpoints of the can be colocated on the same PE router or on different PE routers.

Options

local—Unique local VPWS service identifier of the EVPN-VPWS network. This identifies the logical interface of the PE router forwarding traffic to the PE router with a remote VPWS service identifier in the EVPN-VPWS network.

Range: 1 through 16,777,215

remote—Unique remote VPWS service identifier of the EVPN-VPWS network. This identifies the logical interface of the PE router receiving the traffic from the PE router with a local VPWS service identifier in the EVPN-VPWS network.

Range: 1 through 16,777,215

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Overview of VPWS with EVPN Signaling Mechanisms on page 666
- EVPN Multihoming Overview on page 29
- Configuring VPWS with EVPN Signaling Mechanisms on page 681
- Example: Configuring VPWS with EVPN Signaling Mechanisms on page 683
- **show evpn vpws-instance on page 1193**

vrf-export

Syntax

```
vrf-export [ policy-names ];
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
  vpls mesh-group mesh-group-name]
[edit routing-instances routing-instance-name]
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name]
[edit switch-options]
```

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.
Statement introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.

Description

Specify how routes are exported from the local PE router' s VRF table (*routing-instance-name*.inet.0) to the remote PE router. If the value **vrf** is specified for the **instance-type** statement included in the routing instance configuration, this statement is required.

You can configure multiple export policies on the PE router or PE switch.

Default

If the instance-type is **vrf**, **vrf-export** is a required statement. The default action is to reject.

Options

policy-names— Names for the export policies.

Required Privilege Level

routing— To view this statement in the configuration.
routing-control— To add this statement to the configuration.

Related Documentation

- Implementing EVPN-VXLAN for Data Centers on page 272
- instance-type on page 1061
- *Configuring an Export Policy for the PE Router's VRF Table*

vrf-import

Syntax

```
vrf-import [ policy-names ];
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
  vpls mesh-group mesh-group-name]
[edit routing-instances routing-instance-name]
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name]
[edit switch-options]
```

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.
Statement introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.

Description

Specify how routes are imported into the virtual routing and forwarding (VRF) table (***routing-instance-name***.inet.0) of the local provider edge (PE) router or switch from the remote PE router. If the value **vrf** is specified for the **instance-type** statement included in the routing instance configuration, this statement is required.

You can configure multiple import policies on the PE router or switch.

Default

If the instance type is **vrf**, **vrf-import** is a required statement. The default action is to accept.

Options

policy-names—Names for the import policies.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Implementing EVPN-VXLAN for Data Centers on page 272
- **instance-type on page 1061**
- *Configuring an Import Policy for the PE Router's VRF Table*

vrf-target

Syntax

```
vrf-target {
  community;
  auto
  import community-name;
  export community-name;
}
```

Hierarchy Level

```
[edit logical-systems logical-system-name routing-instances routing-instance-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
  l2vpn mesh-group mesh-group-name],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
  vpls mesh-group mesh-group-name],
[edit routing-instances routing-instance-name protocols evpn vni-options],
[edit routing-instances routing-instance-name protocols l2vpn mesh-group
  mesh-group-name],
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name],
[edit switch-options]
```

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.
Statement introduced in Junos OS Release 14.1X53-D30 for QFX Series switches. **auto option** was also added at this time.

Description

Specify a virtual routing and forwarding (VRF) target community. If you configure the ***community*** option only, default VRF import and export policies are generated that accept and tag routes with the specified target community. The purpose of the **vrf-target** statement is to simplify the configuration by allowing you to configure most statements at the **[edit routing-instances]** hierarchy level. In effect, this statement configures a single policy for import and a single policy for export to replace the per-VRF policies for every community.

You can still create more complex policies by explicitly configuring VRF import and export policies using the **import** and **export** options.

Options

community—Community name.

auto—Automatically derives the route target (RT) for QFX5100 switches.

import *community-name*—Allowed communities accepted from neighbors.

export *community-name*—Allowed communities sent to neighbors.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring a VRF Target*
- *Example: Configuring FEC 129 BGP Autodiscovery for VPWS*

CHAPTER 30

VXLAN Configuration Statements

decapsulate-inner-vlan

Syntax decapsulate-inner-vlan

Hierarchy Level [edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit vlans *vlan-name* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.
Statement introduced in Junos OS Release 17.3R1 for EX9200 switches.

Description Configure the switch to de-encapsulate a preserved original VLAN tag (in the inner Ethernet packet) from a VXLAN encapsulated packet.

Default A preserved VLAN tag is dropped when the packet is de-encapsulated.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding VXLANs on page 260
- *Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- *Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- encapsulate-inner-vlan on page 1097

encapsulate-inner-vlan

Syntax

```
encapsulate-inner-vlan
```

Hierarchy Level

```
[edit routing-instances routing-instance-name vlans vlan-name vxlan],
[edit vlans vlan-name vxlan]
```

Release Information

Statement introduced in Junos OS Release 14.1R2 for MX Series Routers.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series.
Statement introduced in Junos OS Release 17.3R1 for EX9200 switches.

Description

Configure the switch to preserve the original VLAN tag (in the inner Ethernet packet) when performing Virtual Extensible LAN (VXLAN) encapsulation.

Default

The original tag is dropped when the packet is encapsulated.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding VXLANs on page 260
- *Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- *Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- *decapsulate-accept-inner-vlan*

encapsulation vxlan

Syntax

```
encapsulation vxlan;
```

Hierarchy Level

```
{edit protocols evpn],
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 16.1.
Statement introduced in Junos OS Release 17.3R1 for EX Series switches.

Description

Configure a VXLAN encapsulation type. This statement is required for a VXLAN EVPN instance.

NOTE: If you configure the **encapsulation vxlan** statement, then you must also configure the extended-vni-list statement.

NOTE: The **encapsulation vxlan** statement is an exclusive command. You cannot configure the **encapsulation vxlan** statement with the extended-vlan-list statement, or other commands associated with MPLS EVPN instances.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches on page 270
- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373

extended-vni-all

Syntax

```
extended-vni-all;
```

Hierarchy Level

For QFX5100 and EX4600 switches:

```
{edit protocols evpn]
```

For MX Series routers:

```
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 14.1X53-D30 on QFX Series switches.
Statement introduced in Junos OS Release 16.1R1.
Statement introduced in Junos OS Release 18.2R1 on EX Series switches.

Description

Include all VXLAN Network Identifiers (VNIs) as part of the Virtual Switch (VS) instance. By specifying **all**, you bypass the **commit check** process and all configured bridge domains (BDs) in the Ethernet Virtual Private Network (EVPN) are considered extended.

NOTE: The extended-vni-list statement is an exclusive command. You cannot configure the extended-vni-list statement with either the extended-vlan-list or extended-vni-all statements.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373
- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches on page 270
- **extended-vni-list on page 1100**

extended-vni-list

Syntax

```
extended-vni-list [list of VNIs | all];
```

Hierarchy Level

For MX Series routers, EX9200, and QFX Series switches:

```
[edit logical-systems logical-system-name routing-instances routing-instance-name
  protocols],
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 14.1X53-D15 for QFX Series switches.
Statement introduced in Junos OS Release 16.1 for MX Series.
Statement introduced in Junos OS Release 17.3R1 for EX Series switches.
Support for logical systems on MX Series routers added in Junos OS Release 17.4R1.

Description

Establishes which VXLAN Network Identifiers (VNI) will be part of the Virtual Switch (VS) instance. When you issue the **commit check** command to verify the candidate configuration syntax without committing it, it also checks if the specified VNI(s) is associated with a bridge domain (BD) (**bridge-domain** *name* **vxlan vni**).

There are different broadcast, unknown unicast, and multicast (BUM) replication options available in Ethernet Virtual Private Network (EVPN). By using the **extended-vni-list** statement, you forgo a multicast underlay in favor of EVPN and VXLAN ingress-replication.

NOTE: The **extended-vni-list** statement is an exclusive command. You cannot configure the **extended-vni-list** statement with either the extended-vlan-list or extended-vni-all statements.

NOTE: If you configure the **extended-vni-list** statement, then you must also configure the encapsulation vxlan statement.

Options

list of VNIs—Specify a single VNI or list of VNIs as part of the VS instance, for example **extended-vni-list [10-50 60 70]**.

all—Include all VNIs as part of the VS instance. By specifying **all**, you bypass the **commit check** process and all configured BDs in the EVPN are considered extended.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373
- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- EVPN Over VXLAN Encapsulation Configuration Overview for QFX Series and EX4600 Switches on page 270
- *show configuration protocols evpn*
- Implementing EVPN-VXLAN for Data Centers on page 272

ingress-node-replication (EVPN)

Syntax

```
ingress-node-replication;
```

Hierarchy Level

[edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit vlans *vlan-name* vxlan]

Release Information

Statement introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.
Statement introduced in Junos OS Release 17.3R1 for EX9200 switches.

Description

Ingress replication is supported only for EVPN-VXLAN and enabled by default; no configuration is needed. EVPN A/A with VXLAN encapsulation is based on the local bias for traffic coming from the access layer (redundant Layer 2 gateway function through a pair of top-of-rack switches). Because the traffic has no MPLS label, the split-horizon filtering rule for multi-home Ethernet segment is modified to be based on the IP address of the EVPN provider edge (PE) instead of the MPLS ES-label. This is called local bias for EVPN-VXLAN. Each EVPN PE tracks the IP address of its peer multihomed EVPN PE that share the same Ethernet segment. This is the source VTEP IP address (outer SIP) for each VXLAN packet received from other EVPN PE. The local bias filtering rule is enforced on both ingress and egress PEs for the multidestination traffic. For egress traffic, there is no forwarding of any multidestination packets to the same multihomed Ethernet segment that an egress PE shares with its ingress PE regardless of the egress PE's DF election status for that Ethernet segment. Ingress traffic is responsible for forwarding multi-destination packets coming from any directly attached access interfaces to the rest of the multi-home Ethernet segments associated with it regardless of the ingress PE's designated forwarder election status on the connected physical device segment.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247

interface-num

Syntax interface-num *integer*;

Hierarchy Level [edit forwarding-options vxlan-routing]

Release Information Statement introduced in Junos OS Release 17.3R1 for QFX Series switches.

Description On a QFX5110 switch, configure the maximum number of physical interfaces reserved for use in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) overlay network.

The QFX5110 switch supports a maximum of 12,288 physical interfaces for use in either underlay networks or an EVPN-VXLAN overlay network. By default, the switch allocates 4000 physical interfaces for use in an EVPN-VXLAN network, which leaves 8288 physical interfaces for use in underlay networks. The **interface-num** configuration statement enables you to re-allocate the maximum number of physical interfaces reserved for use in an EVPN-VXLAN network.

If you do not plan to use a QFX5110 switch in an EVPN-VXLAN network, you can set the **interface-num** configuration statement to 0, which allocates all 12,288 physical interfaces for use in underlay networks.

Conversely, if you plan to use a QFX5110 switch in an EVPN-VXLAN network, you can re-allocate a greater number of the physical interfaces to the EVPN-VXLAN network.

NOTE: **Changing the default number of physical interfaces reserved for use in an EVPN-VXLAN network causes the Packet Forwarding Engine to restart. Restarting the Packet Forwarding Engine interrupts all forwarding operations. Therefore, we strongly recommend using this configuration statement before the EVPN-VXLAN network becomes operational.**

Options ***integer***—Maximum number of physical interfaces reserved for use in an EVPN-VXLAN overlay network on a QFX5110 switch.

Range: 0 through 12,288. The specified value must be a multiple of 2048—for example, 2048, 4096, 6144, and so on.

Default: 4000

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

next-hop (VXLAN Routing)

Syntax next-hop *integer*;

Hierarchy Level [edit forwarding-options vxlan-routing]

Release Information Statement introduced in Junos OS Release 17.3R1 for QFX Series switches.

Description On a QFX5110 switch, configure the maximum number of next hops reserved for use in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) overlay network.

The QFX5110 switch can store a maximum of 49,152 next hops for use in either underlay networks or an EVPN-VXLAN overlay network. By default, the switch allocates 8000 next hops for use in an EVPN-VXLAN network, which leaves 41,152 next hops for use in underlay networks. The **next-hop** configuration statement enables you to re-allocate the maximum number of next hops reserved for use in an EVPN-VXLAN network.

If you do not plan to use a QFX5110 switch in an EVPN-VXLAN network, you can set the **next-hop** configuration statement to 0, which allocates all 49,152 next hops for use in underlay networks.

Conversely, if you plan to use a QFX5110 switch in an EVPN-VXLAN network, you can re-allocate a greater number of the next hops to the EVPN-VXLAN network.

NOTE: Changing the default number of next hops reserved for use in an EVPN-VXLAN network causes the Packet Forwarding Engine to restart. Restarting the Packet Forwarding Engine interrupts all forwarding operations. Therefore, we strongly recommend using this configuration statement before the EVPN-VXLAN network becomes operational.

Options ***integer***—Maximum number of next hops reserved for use in an EVPN-VXLAN overlay network on a QFX5110 switch.

Range: 0 through 49,152. The specified value must be a multiple of 4096—for example, 4096, 8192, 12.288, and so on.

Default: 8000

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

virtual-gateway-address

Syntax virtual-gateway-address *address*;

Hierarchy Level [edit interfaces *interface-name* unit *logical-unit-number* family *family* address *address*]

Release Information Statement introduced in Junos OS Release 16.1.

Description Set the default IPv4 or IPv6 address for the gateway for end hosts. Because you must configure the virtual gateway address using the same IP address as on all of the provider edge (PE) devices (which internally generates the same Virtual Router Redundancy Protocol (VRRP) MAC), the need to proxy for remote gateway IP addresses is eliminated. Every logical integrated routing and bridging (IRB) interface can have a corresponding virtual gateway address. The maximum number of PEs that can have the same virtual gateway address is 64.

To support pinging on the virtual gateway IP address, you must include both the **virtual-gateway-accept-data** statement and the **preferred** statement at the [**edit interfaces irb unit**] hierarchy of the preferred virtual gateway.

Options ***address***—virtual gateway address. You cannot specify the addresses 0.0.0.0 (default route address) or 255.255.255.255 (broadcast IP address).

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- EVPN with IRB Solution Overview on page 600
- Example: Configuring EVPN with IRB Solution on page 620
- *Configuring the Interface Address*

virtual-gateway-v4-mac

Syntax virtual-gateway-v4-mac *ipv4-mac-address*

Hierarchy Level [edit dynamic-profiles *name* interfaces *name* unit *logical-unit-number*],
[edit dynamic-profiles *name* logical-systems *name* interfaces *name* unit *logical-unit-number*],
[edit interfaces *name* unit *logical-unit-number*]

Release Information Statement introduced in Junos OS Release 14.2R5.
Statement introduced in Junos OS Release 15.1X53-D63 for QFX Series switches.

Description Explicitly configure an IPv4 media access control (MAC) address for a default virtual gateway.

A default virtual gateway is created when you specify a virtual gateway address (VGA) while configuring an integrated routing and bridging (IRB) interface on a Juniper Networks device that functions as a Layer 3 Virtual Extensible LAN (VXLAN) gateway in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) topology with a two-layer IP fabric. Through the IRB interface with which it is configured, the default virtual gateway enables communication between non-virtualized hosts, virtual machines (VMs), and servers in different VXLANs or IP subnetworks.

When you configure a VGA for an IRB interface, the Layer 3 VXLAN gateway automatically generates IPV4 MAC address 00:00:5e:00:01:01 for that particular virtual gateway. (This topic refers to the virtual gateway MAC address as a virtual MAC.) The automatically generated virtual MAC is not included as the source MAC address in packets generated by the Layer 3 VXLAN gateway. Instead, data packets and the source MAC address field in the outer Ethernet header of Address Resolution Protocol (ARP) replies and neighbor advertisement packets include the MAC address for the IRB interface. (This topic refers to the MAC address for the IRB interface as the IRB MAC.)

When an ARP reply includes the IRB MAC as the source MAC address instead of the virtual MAC, an issue might arise in an EVPN-VXLAN topology with a two-layer IP fabric. This issue might result in the flooding of unknown-unicast packets throughout the domain.

If you explicitly configure a MAC address for a default virtual gateway, the automatically generated virtual MAC is overridden by the configured virtual MAC. That is, when the Layer 3 VXLAN gateway sends data packets, ARP replies, and neighbor advertisement packets, the configured virtual MAC is in the outer Ethernet header of these packets. As a result, the possibility that the domain is flooded with unknown-unicast packets is eliminated.

For more information about the flooding issue and its resolution, see "Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Topology" on page 298.

Options ***ipv4-mac-address***—IPv4 MAC address for the default virtual gateway.

Required Privilege Level

admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- virtual-gateway-v6-mac on page 1107

virtual-gateway-v6-mac

Syntax virtual-gateway-v6-mac *ipv6-mac-address*

Hierarchy Level [edit dynamic-profiles *name* interfaces *name* unit *logical-unit-number*],
[edit dynamic-profiles *name* logical-systems *name* interfaces *name* unit *logical-unit-number*],
[edit interfaces *name* unit *logical-unit-number*]

Release Information Statement introduced in Junos OS Release 14.2R5.
Statement introduced in Junos OS Release 15.1X53-D63 for QFX Series switches.

Description Explicitly configure an IPv6 media access control (MAC) address for a default virtual gateway.

A default virtual gateway is created when you specify a virtual gateway address (VGA) while configuring an integrated routing and bridging (IRB) interface on a Juniper Networks device that functions as a Layer 3 Virtual Extensible LAN (VXLAN) gateway in an Ethernet VPN-Virtual Extensible LAN (EVPN-VXLAN) topology with a two-layer IP fabric. Through the IRB interface with which it is configured, the default virtual gateway enables communication between non-virtualized hosts, virtual machines (VMs), and servers in different VXLANs or IP subnetworks.

When you configure a VGA for an IRB interface, the Layer 3 VXLAN gateway automatically generates IPv6 MAC address 00:00:5E:00:02:01 for that particular virtual gateway. (This topic refers to the virtual gateway MAC address as a virtual MAC.) The automatically generated virtual MAC is not included as the source MAC address in packets generated by the Layer 3 VXLAN gateway. Instead, data packets and the source MAC address field in the outer Ethernet header of Address Resolution Protocol (ARP) replies and neighbor advertisement packets include the MAC address for the IRB interface. (This topic refers to the MAC address for the IRB interface as the IRB MAC.)

When an ARP reply includes the IRB MAC as the source MAC address instead of the virtual MAC, an issue might arise in an EVPN-VXLAN topology with a two-layer IP fabric. This issue might result in the flooding of unknown-unicast packets throughout the domain.

If you explicitly configure a MAC address for a default virtual gateway, the automatically generated virtual MAC is overridden by the configured virtual MAC. That is, when the Layer 3 VXLAN gateway sends data packets, ARP replies, and neighbor advertisement packets, the configured virtual MAC is in the outer Ethernet header of these packets. As a result, the possibility that the domain is flooded with unknown-unicast packets is eliminated.

For more information about the flooding issue and its resolution, see "Understanding the MAC Addresses For a Default Virtual Gateway in an EVPN-VXLAN Topology" on page 298.

Options ***ipv6-mac-address***—IPv6 MAC address for the default virtual gateway.

Required Privilege Level
admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation
- virtual-gateway-v4-mac on page 1105

vni

Syntax
```
vni [1–16777214]
```

Syntax (MX Series)
```
vni [0–16777214]
```

Hierarchy Level
```
[edit routing-instances routing-instance-name vlans vlan-name vxlan]
[edit vlans vlan-name vxlan]
```

Hierarchy Level (MX Series)
```
[edit interfaces name unit name tunnel encapsulation vxlan-gpe]
```

Release Information
Statement introduced in Junos OS Release 14.1R2 for MX Series routers.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 17.3R1 for EX9200 switches.
Statement introduced in Junos OS Release 18.3R1 for MX Series routers.

Description
Assign a numeric value to identify a Virtual Extensible LAN (VXLAN). All members of a VXLAN must use the same VNI.

Options
vni—Value to write to the vni field
Range: 0 through 16777215
Range: (MX Series routers) 0 through 16777214

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Understanding VXLANs on page 260
- *Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- *Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches*

vni-options

Syntax

```
vni-options vni vxlan-network-identifier {
  designated-forwarder-election-hold-time seconds;
  vrf-target {
    community;
    auto;
    import community-name;
    export community-name;
  }
}
```

Hierarchy Level

```
[edit protocols evpn]
[edit routing-instances routing-instance-name protocols evpn]
```

Release Information

Statement introduced in Junos OS Release 14.1X53-D30 for QFX Series switches.
Statement introduced in Junos OS Release 16.1 for MX Series.
Statement introduced in Junos OS Release 17.3R1 for EX Series switches.

Description

Configure a designated forwarder election hold time and specific route targets (RTs) for each VXLAN network identifier (VNI).

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- EVPN Multihoming Overview on page 29
- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373
- **extended-vni-list on page 1100**

vnid (EVPN)

Syntax

```
vnid;
```

Hierarchy Level

```
[edit routing-instances routing-instance-name ],
```

Release Information

Statement introduced in Junos OS Release 17.1 on MX Series routers with MPCs.

Description

Enables having the same MAC frames across multiple VXLAN segments without traffic crossover. Multicast in VXLAN is implemented as Layer 3 multicast, in which endpoints subscribe to groups. VNID is a 24-bit virtual network identifier that uniquely identifies the VXLAN segment.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding EVPN with VXLAN Data Plane Encapsulation on page 247
- Example: Configuring an EVPN Control Plane and VXLAN Data Plane on page 373
- EVPN Type-5 Route with MPLS encapsulation for EVPN-MPLS on page 14
- EVPN Type-5 Route with VXLAN encapsulation for EVPN-VXLAN on page 13

vxlan

Syntax

```
vxlan {
  encapsulate-inner-vlan;
  ingress-node-replication;
  multicast-group;
  ovsdb-managed;
  unreachable-vtep-aging-timer
  vni;
  multicast-group multicast-group;
  }
}
```

Hierarchy Level

```
[edit vlans]
[edit bridge-domains bridge-domain-name]
```

Release Information

Statement introduced in Junos OS Release 14.1X53-D10.
ingress-node-replication option added for EVPN VXLAN on QFX5100 switches in Junos OS Release 14.1X53-D30.
multicast-group*multicast-group* option added for MX series routers with MPC and MIC interfaces in Junos OS Release 17.2.

Description

Configure support for Virtual Extensible LANs (VXLANs) on a Juniper Networks device.

Options

multicast-group *mulicast-group*—Multicast group (IPv4 or IPv6 addresses) registered for VXLAN segment.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding VXLANs on page 260
- *Manually Configuring VXLANs on QFX Series and EX4600 Switches*
- *Examples: Manually Configuring VXLANs on QFX Series and EX4600 Switches*

vxlan-routing

Syntax

```
vxlan-routing {
  interface-num integer;
  next-hop integer;
  overlay-ecmp;
}
```

Hierarchy Level

```
[edit forwarding-options]
```

Release Information

Statement introduced in Junos OS Release 17.3R1 for QFX Series switches.
overlay-ecmp statement introduced in Junos OS Release 17.4R1 for QFX5110 switches.

Description

Configure Virtual Extensible LAN (VXLAN) routing options on a QFX5110 switch.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

CHAPTER 31

Operational Commands

- clear ethernet-switching evpn nd-statistics
- clear ethernet-switching evpn nd-table
- clear evpn duplicate-mac-suppression
- clear evpn nd-table
- clear evpn statistics
- show ethernet-switching evpn nd-statistics
- show ethernet-switching evpn nd-table
- show ethernet-switching flood
- show ethernet-switching table
- show evpn arp-table
- show evpn database
- show evpn flood
- show evpn instance
- show evpn ip-prefix-database
- show evpn l3-context
- show evpn mac-table
- show evpn nd-table
- show evpn p2mp
- show evpn peer-gateway-macs
- show evpn prefix
- show evpn vpws-instance
- show route forwarding-table
- show route table
- show vlans evpn nd-table

clear ethernet-switching evpn nd-statistics

Syntax clear ethernet-switching evpn nd-statistics

Release Information Command introduced in Junos OS Release 17.3R1 for EX series switches.

Description Clear the Neighbor Discovery (ND) proxy table statistics for integrated routing and bridging (IRB) interfaces participating in the Ethernet VPN (EVPN). ND proxy is a kernel module that implements IPv6 Neighbor Discovery proxying over Ethernet-like access networks.

Options **none**—Clear the ND proxy table statistics for IRB interfaces participating in the EVPN.

Required Privilege Level clear

List of Sample Output clear ethernet-switching evpn nd-statistics on page 1114

Output Fields When you enter this command, the ND proxy table statistics for IRB interfaces participating in the EVPN are cleared.

Sample Output

clear ethernet-switching evpn nd-statistics

```
user@host> clear ethernet-switching evpn nd-statistics
```

clear ethernet-switching evpn nd-table

Syntax clear ethernet-switching evpn nd-table

Release Information Command introduced in Junos OS Release 17.3R1 for the EX Series.

Description Clear the Neighbor Discovery (ND) proxy table from the Ethernet VPN (EVPN) for integrated routing and bridging (IRB) interfaces. This command applies to EVPN instances of type **virtual-switch**.

Options **none**—Clear information about the ND proxy tables for an EVPN.

Required Privilege Level clear

List of Sample Output clear ethernet-switching evpn nd-table on page 1115

Output Fields When you enter this command, you are provided feedback on the status of your request.

Sample Output

clear ethernet-switching evpn nd-table

```
user@host> clear ethernet-switching evpn nd-table
```

clear evpn duplicate-mac-suppression

Syntax

```
clear evpn duplicate-mac-suppression
<instance instance |l2-domain-id l2-domain-id | logical-system (all |logical-system-name)
  | mac-address mac-address>
```

Release Information Command introduced in Junos OS Release 17.4 on MX Series routers, EX Series switches, and QFX Series switches.

Description Clear suppressed duplicate MAC address in the EVPN network.

Options

instance *instance*—(Optional) Clear suppressed MAC address for a specific routing instance. On MX Series routers , the routing instance can be an EVPN instance or virtual-switch instance. On QFX Series switches, the routing instance can be an EVPN instance, virtual-switch instance or an implicit default-switch instance.

l2-domain-id *l2-domain-id*—(Optional) Clear suppressed MAC address for a specific L2 domain.

logical-system (all | *logical-system-name*)—(Optional) Clear suppressed MAC address on all logical systems or on a specific logical system.

mac-address *mac-address*—(Optional) Clear a specific suppressed MAC address.

Required Privilege Level maintenance

Related Documentation

- Overview of MAC Mobility on page 8
- Changing Duplicate MAC Address Detection Settings on page 11

clear evpn nd-table

Syntax clear evpn nd-table

Release Information Command introduced in Junos OS Release 16.2.
Command introduced in Junos OS Release 17.1 for MX series Routers.
Command introduced in Junos OS Release 17.3R1 for EX series switches.

Description Clear the Neighbor Discovery (ND) proxy table from the Ethernet VPN (EVPN). ND proxy is a kernel module that implements IPv6 Neighbor Discovery proxying over Ethernet-like access networks.

Options **none**—Clear learned ND proxy tables from the EVPN.

Required Privilege Level clear

List of Sample Output clear evpn nd-table on page 1117

Output Fields When you enter this command, you are provided feedback on the status of your request.

Sample Output

clear evpn nd-table

```
user@host> clear evpn nd-table
```

clear evpn statistics

Syntax

```
clear evpn statistics instance evpn-instance
<logical-system (all | logical-system-name)>
```

Release Information Command introduced in Junos OS Release 17.2 on MX Series routers.

Description Clear EVPN statistics of all interfaces in a routing instance of type EVPN.

Options ***evpn-instance***—Name of the EVPN routing instance.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level maintenance

Related Documentation

- *clear bridge statistics*

List of Sample Output clear evpn statistics on page 1118

Output Fields This command produces no output.

Sample Output

clear evpn statistics

```
user@host> clear evpn statistics instance evpn-instance
```

show ethernet-switching evpn nd-statistics

Syntax

```
show ethernet-switching evpn nd-statistics
<brief |count | detail | extensive | summary | display xml>
```

Release Information Command introduced in Junos OS Release 17.3R1 for EX series switches.

Description Show Neighbor Discovery (ND) proxy table statistics for integrated routing and bridging (IRB) interfaces participating in the Ethernet VPN (EVPN). ND proxy is a kernel module that implements IPv6 Neighbor Discovery proxying over Ethernet-like access networks.

Options

none—Display information about ND proxy table statistics for all IRB interfaces participating in the EVPN.

brief | count | detail | extensive |summary | display xml—(Optional) Display the specified level of output.

Required Privilege Level view

List of Sample Output show ethernet-switching evpn nd-statistics on page 1120

Output Fields Table 27 on page 1119 lists the output fields for the **show ethernet-switching evpn nd-statistics** command. Output fields are listed in the approximate order in which they appear.

Table 27: show ethernet-switching evpn nd-statistics Output Fields

Field Name	Field Description	Level of Output
Interface	IRB interface on which the statistics has been applied	All levels
EVPN	Name of the EVPN	All levels
Bridge domain	Name of the bridge domain for the EVPN	All levels
ND routes add received	Total number of additional ND routes received	All levels
ND routes del received	Total number of deleted ND routes received	All levels
ND routes dropped	Total number of ND routes dropped	All levels
MAC+IPv6s sent to peer	Total number of MAC and IPv6 addresses sent to peer device	All levels

Sample Output

show ethernet-switching evpn nd-statistics

```
user@host> show ethernet-switching evpn nd-statistics interface irb.0


Interface : irb.0       EVPN : evpn1    Bridge domain: vlan10
ND routes add received          : 2
ND routes del received          : 0
ND routes dropped               : 0
MAC+IPv6s sent to peer          : 3
```

show ethernet-switching evpn nd-table

Syntax

```
show ethernet-switching evpn nd-table
<brief | detail | extensive>
<count>
<instance-name>
<mac-address>
<vlan-name>
```

Release Information Command introduced in Junos OS Release 17.4R2 for EX Series Switches.

NOTE: Starting with Junos OS Release 17.4R2, the **show ethernet-switching evpn nd-table** command replaces the show vlans evpn nd-table command.

Description Display information about INET entries associated with MAC addresses learned through Network Discovery Protocol (NDP).

Options

none—Display information for all INET entries.

brief | detail | extensive—(Optional) Display the specified level of output.

count—(Optional) Display the number of INET addresses learned in a routing instance.

instance—(Optional) Display information for a specified instance.

mac-adress—(Optional) Display information for a specified MAC address.

vlan-name (all)—(Optional) Display information for a specified VLAN or for all VLANs.

Required Privilege Level view

List of Sample Output

Output Fields Table 28 on page 1122 lists the output fields for the **show ethernet-switching evpn nd-table** command. Output fields are listed in the approximate order in which they appear.

Table 28: show ethernet-switching evpn nd-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the NDP table.	All levels
MAC address	MAC addresses learned through NDP.	**brief**, **detail**, **extensive**, **instance**, **mac-addressvlan-name**,,
Logical Interface	Logical interface associated with the routing instance in which the NDP INET address is learned.	**brief**, **instance**, **mac-addressvlan-name**,,
Routing instance	Routing instance in which the NDP INET address is learned.	all levels
Bridging domain	Bridging domain in which the NDP INET address is learned.	all levels
Learning interface	Interface on which the NDP INET address is learned.	**detail**, **extensive**
Count	Indicates the number of NDP INET addresses learned in a routing instance in a bridge domain.	**count**

Sample Output

show ethernet-switching evpn nd-table brief

```
user@switch> show ethernet-switching evpn nd-table brief

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2            00:05:86:a0:d5:00 irb.0         evpn1      vlan10
```

show ethernet-switching evpn nd-table detail

```
user@switch> show ethernet-switching evpn nd-table detail

INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
   Bridging domain: vlan10
   Learning interface: irb.0
```

show ethernet-switching evpn nd-table extensive

```
user@switch> show ethernet-switching evpn nd-table extensive

INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
   Bridging domain: vlan10
   Learning interface: irb.0
```

show ethernet-switching evpn nd-table count

```
user@switch> show ethernet-switching evpn nd-table count

1 ND INET addresses learned in routing instance evpn1 bridge domain vlan10
```

show ethernet-switching evpn nd-table instance evpn1

```
user@switch> show ethernet-switching evpn nd-table instance evpn1

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2
           00:05:86:a0:d5:00
  irb.0         evpn1        vlan10
```

show ethernet-switching evpn nd-table instance 00:05:86:90:bd:f0 (MAC address)

```
user@switch> show ethernet-switching evpn nd-table

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2
           00:05:86:a0:d5:00
  irb.0         evpn1        __evpn1__
```

show ethernet-switching evpn nd-table vlan-name vlan10

```
user@switch>show ethernet-switching evpn nd-table vlan-name vlan10


INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2           00:05:86:a0:d5:00
  irb.0         evpn1        vlan10
```

show ethernet-switching flood

Syntax

```
show ethernet-switching flood
<brief | detail | extensive>
<event-queue>
<instance instance-name>
<logical-system logical-system-name>
<route (all-ce-flood | all ve-flood | alt-root-flood | bd-flood | mlp-flood | re-flood)>
<vlan-name vlan-name>
```

Release Information

Command introduced in Junos OS Release 12.3R2.
Command introduced in Junos OS Release 12.3R2 for EX Series switches.
Command introduced in Junos OS Release 17.4R1 for QFX Series switches.

Description

(EX Series switches and QFX Series switches only) Display Ethernet-switching flooding information.

Options

none—Display all Ethernet-switching flooding information for all VLANs.

brief | detail | extensive—(Optional) Display the specified level of output.

event-queue—(Optional) Display the queue of pending Ethernet-switching flood events.

instance *instance-name*—(Optional) Display Ethernet-switching flooding information for the specified routing instance.

logical-system *logical-system-name*—(Optional) Display Ethernet-switching flooding information for the specified logical system.

route (all-ce-flood | all ve-flood | alt-root-flood | bd-flood | mlp-flood | re-flood)—(Optional) Display the following:

- **all-ce-flood**—Display the route for flooding traffic to all customer edge routers or switches if **no-local-switching** is enabled.
- **all-ve-flood**—Display the route for flooding traffic to all VPLS edge routers or switches if **no-local-switching** is enabled.
- **alt-root-flood**—Display the Spanning Tree Protocol (STP) alt-root flooding route used for the interface.
- **bd-flood**—Display the route for flooding traffic of a VLAN if **no-local-switching** is not enabled.
- **mlp-flood**—Display the route for flooding traffic to MAC learning chips.
- **re-flood**—Display the route for Routing Engine flooding to all interfaces.

vlan-name *vlan-name*—(Optional) Display Ethernet-switching flooding information for the specified VLAN.

Required Privilege Level view

List of Sample Output

Sample Output

show ethernet-switching flood

```
user@host> show ethernet-switching flood

Name: __juniper_private1__
CEs: 0
VEs: 0
Name: default-switch
CEs: 9
VEs: 0
VLAN Name: VLAN101
Flood Routes:
  Prefix    Type          Owner                NhType          NhIndex
  0x3057b/51 FLOOD_GRP_COMP_NH __all_ces__     comp            12866
  0x30004/51 FLOOD_GRP_COMP_NH __re_flood__    comp            12863
VLAN Name: VLAN102
Flood Routes:
  Prefix    Type          Owner                NhType          NhIndex
  0x3057c/51 FLOOD_GRP_COMP_NH __all_ces__     comp            12875
  0x30005/51 FLOOD_GRP_COMP_NH __re_flood__    comp            12872
VLAN Name: VLAN103
Flood Routes:
  Prefix    Type          Owner                NhType          NhIndex
  0x3057d/51 FLOOD_GRP_COMP_NH __all_ces__     comp            12884
  0x30006/51 FLOOD_GRP_COMP_NH __re_flood__    comp            12881
```

show ethernet-switching flood brief

```
user@host> show ethernet-switching flood brief

Name                Active CEs         Active VEs
__juniper_private1__ 0                 0
default-switch      9                  0
```

show ethernet-switching flood detail

```
user@host> show ethernet-switching flood detail

Name: __juniper_private1__
CEs: 0
VEs: 0
Name: default-switch
CEs: 9
```

```
VEs: 0
VLAN Name: VLAN101
Flood Routes:
  Prefix    Type          Owner                NhType          NhIndex
  0x3057b/51 FLOOD_GRP_COMP_NH __all_ces__     comp            12866
  0x30004/51 FLOOD_GRP_COMP_NH __re_flood__    comp            12863
VLAN Name: VLAN102
Flood Routes:
  Prefix    Type          Owner                NhType          NhIndex
  0x3057c/51 FLOOD_GRP_COMP_NH __all_ces__     comp            12875
  0x30005/51 FLOOD_GRP_COMP_NH __re_flood__    comp            12872
VLAN Name: VLAN103
Flood Routes:
  Prefix    Type          Owner                NhType          NhIndex
  0x3057d/51 FLOOD_GRP_COMP_NH __all_ces__     comp            12884
  0x30006/51 FLOOD_GRP_COMP_NH __re_flood__    comp            12881
```

show ethernet-switching flood extensive

user@host> **show ethernet-switching flood extensive**

```
Name: __juniper_private1__
CEs: 0
VEs: 0
Name: default-switch
CEs: 9
VEs: 0
VLAN Name: VLAN101
  Flood route prefix: 0x3057b/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __all_ces__
  Flood group name: __all_ces__
  Flood group index: 1
  Nexthop type: comp
  Nexthop index: 12866
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            12860
        Composition: split-horizon
        Flooding to:
        Name             Type          NhType          Index
        ae20.0           CE             ucst            7605

  Flood route prefix: 0x30004/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __re_flood__
  Flood group name: __re_flood__
  Flood group index: 65534
  Nexthop type: comp
  Nexthop index: 12863
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            12860
        Composition: split-horizon
        Flooding to:
        Name             Type          NhType          Index
        ae20.0           CE             ucst            7605
VLAN Name: VLAN102
```

```
  Flood route prefix: 0x3057c/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __all_ces__
  Flood group name: __all_ces__
  Flood group index: 1
  Nexthop type: comp
  Nexthop index: 12875
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            12869
        Composition: split-horizon
        Flooding to:
        Name             Type          NhType          Index
        ae20.0           CE             ucst            7605

  Flood route prefix: 0x30005/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __re_flood__
  Flood group name: __re_flood__
  Flood group index: 65534
  Nexthop type: comp
  Nexthop index: 12872
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            12869
        Composition: split-horizon
        Flooding to:
        Name             Type          NhType          Index
        ae20.0           CE             ucst            7605
VLAN Name: VLAN103

  Flood route prefix: 0x3057d/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __all_ces__
  Flood group name: __all_ces__
  Flood group index: 1
  Nexthop type: comp
  Nexthop index: 12884
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            12878
        Composition: split-horizon
        Flooding to:
        Name             Type          NhType          Index
        ae20.0           CE             ucst            7605

  Flood route prefix: 0x30006/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __re_flood__
  Flood group name: __re_flood__
  Flood group index: 65534
  Nexthop type: comp
  Nexthop index: 12881
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            12878
        Composition: split-horizon
        Flooding to:
```

```
       Name             Type          NhType          Index
       ae20.0           CE             ucst            7605
VLAN Name: VLAN104
```

show ethernet-switching flood extensive (Junos Fusion Data Center with EVPN)

```
user@host> show ethernet-switching flood extensive
```

```
Name: __juniper_private1__
CEs: 0
VEs: 0
Name: default-switch
CEs: 3
VEs: 3
VLAN Name: v100
  Flood route prefix: 0x3001b/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __ves__
  Flood group name: __ves__
  Flood group index: 0
  Nexthop type: comp
  Nexthop index: 1946
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            1945
        Composition: split-horizon
        Flooding to:
        Name             Type          NhType          Index
        ae0.0            CE             ucst            1886

  Flood route prefix: 0x3000f/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __all_ces__
  Flood group name: __all_ces__
  Flood group index: 1
  Nexthop type: comp
  Nexthop index: 1905
    Flooding to:
    Name             Type          NhType          Index
    __ves__          Group          comp            1971
        Composition: flood-to-all
        Flooding to:
        Name             Type          NhType          Index
        vtep.32769       CORE_FACING    venh            1917
        vtep.32770       CORE_FACING    venh            1918
        vtep.32771       CORE_FACING    venh            1923
    Flooding to:
    Name             Type          NhType          Index
    __all_ces__      Group          comp            1945
        Composition: split-horizon
        Flooding to:
        Name             Type          NhType          Index
        ae0.0            CE             ucst            1886

  Flood route prefix: 0x30001/51
  Flood route type: FLOOD_GRP_COMP_NH
  Flood route owner: __re_flood__
  Flood group name: __re_flood__
```

```
Flood group index: 65534
Nexthop type: comp

      Name             Type          NhType          Index
      vtep.32769       CORE_FACING    venh            1917
      vtep.32770       CORE_FACING    venh            1918
      vtep.32771       CORE_FACING    venh            1923
  Flooding to:
  Name             Type          NhType          Index
  __all_ces__      Group          comp            1907
      Composition: split-horizon
      Flooding to:
      Name             Type          NhType          Index
      ae12.0           CE             ucst            1681

Flood route prefix: 0x30006/51
Flood route type: FLOOD_GRP_COMP_NH
Flood route owner: __re_flood__
Flood group name: __re_flood__
Flood group index: 65534
Nexthop type: comp
Nexthop index: 1891
  Flooding to:
  Name             Type          NhType          Index
  __ves__          Group          comp            1961
      Composition: flood-to-all
      Flooding to:
      Name             Type          NhType          Index
      vtep.32769       CORE_FACING    venh            1917
      vtep.32770       CORE_FACING    venh            1918
      vtep.32771       CORE_FACING    venh            1923
  Flooding to:
  Name             Type          NhType          Index
  __all_ces__      Group          comp            1907
      Composition: split-horizon
      Flooding to:
      Name             Type          NhType          Index
      ae12.0           CE             ucst            1681

...
```

show ethernet-switching table

List of Syntax

Syntax (QFX Series, QFabric, NFX Series and EX4600)

```
show ethernet-switching table
<brief | detail | extensive | summary>
<interface interface-name>
<management-vlan>
<sort-by (name | tag)>
<vlan vlan-name>
```

Syntax (EX Series)

```
show ethernet-switching table
<brief | detail | extensive | summary>
<interface interface-name>
<management-vlan>
<persistent-mac <interface interface-name>>
<sort-by (name | tag)>
<vlan vlan-name>
```

Syntax (EX Series, MX Series and QFX Series)

```
show ethernet-switching table
<brief | count | detail | extensive | summary>
<address>
<instance instance-name>
<interface interface-name>
isid isid
<logical-system logical-system-name>
<persistent-learning (interface interface-name | mac mac-address)>
<address>
<vlan-id (all-vlan | vlan-id)>
<vlan-name (all | vlan-name)>
```

Syntax (SRX Series)

```
show ethernet-switching table (brief |detail |extensive) interface  interface-name
```

Release Information

Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 9.5 for SRX Series.
Options **summary**, **management-vlan**, and **vlan** *vlan-name* introduced in Junos OS Release 9.6 for EX Series switches.
Option **sort-by** and field name **tag** introduced in Junos OS Release 10.1 for EX Series switches.
Command introduced in Junos OS Release 11.1 for the QFX Series.
Output for private VLANs introduced in Junos OS Release 12.1 for the QFX Series.
Option **persistent-mac** introduced in Junos OS Release 11.4 for EX Series switches.
Command introduced in Junos OS Release 12.3R2.

Command introduced in Junos OS Release 12.3R2 for EX Series switches.
Options **logical-system**, **persistent-learning**, and **summary** introduced in Junos OS Release 13.2X50-D10 (ELS).

Description Displays the Ethernet switching table.

(MX Series routers,EX Series switches only) Displays Layer 2 MAC address information.

Options For QFX Series, QFabric, NFX Series and EX4600:

none—(Optional) Display brief information about the Ethernet switching table.

brief | detail | extensive | summary—(Optional) Display the specified level of output.

interface ***interface-name***—(Optional) Display the Ethernet switching table for a specific interface.

management-vlan—(Optional) Display the Ethernet switching table for a management VLAN.

persistent-mac <interface ***interface-name*****>**—(Optional) Display the persistent MAC addresses learned for all interfaces or a specified interface. You can use this command to view entries that you want to clear for an interface that you intentionally disabled.

sort-by (*name* **|** ***tag*****)**—(Optional) Display VLANs in ascending order of VLAN IDs or VLAN names.

vlan ***vlan-name***—(Optional) Display the Ethernet switching table for a specific VLAN.

For EX Series, MX Series and QFX Series:

none—Display all learned Layer 2 MAC address information.

brief | count | detail | extensive | summary—(Optional) Display the specified level of output.

address—(Optional) Display the specified learned Layer 2 MAC address information.

instance ***instance-name***—(Optional) Display learned Layer 2 MAC addresses for the specified routing instance.

interface ***interface-name***—(Optional) Display learned Layer 2 MAC addresses for the specified interface.

isid ***isid***—(Optional) Display learned Layer 2 MAC addresses for the specified ISID.

logical-system ***logical-system-name***—(Optional) Display Ethernet-switching statistics information for the specified logical system.

persistent-learning (interface ***interface-name*** **| mac** ***mac-address*****)**—(Optional) Display dynamically learned MAC addresses that are retained despite device restarts and

interface failures for a specified interface, or information about a specified MAC address.

vlan-id (all-vlan | *vlan-id*)—(Optional) Display learned Layer 2 MAC addresses for all VLANs or for the specified VLAN.

vlan-name (all | *vlan-name*)—(Optional) Display learned Layer 2 MAC addresses for all VLANs or for the specified VLAN.

For SRX Series:

- **none**—(Optional) Display brief information about the Ethernet switching table.
- **brief | detail | extensive**—(Optional) Display the specified level of output.
- **interface-name**—(Optional) Display the Ethernet switching table for a specific interface.

Additional Information

When Layer 2 protocol tunneling is enabled, the tunneling MAC address 01:00:0c:cd:cd:d0 is installed in the MAC table. When the Cisco Discovery Protocol (CDP), Spanning Tree Protocol (STP), or VLAN Trunk Protocol (VTP) is configured for Layer 2 protocol tunneling on an interface, the corresponding protocol MAC address is installed in the MAC table.

Required Privilege Level

view

Related Documentation

- *Example: Setting Up Basic Bridging and a VLAN on Switches*
- *Example: Setting Up Bridging with Multiple VLANs*
- *Example: Setting Up Basic Bridging and a VLAN for an EX Series Switch*
- *Example: Setting Up Bridging with Multiple VLANs for EX Series Switches*
- *Example: Setting Up Q-in-Q Tunneling on EX Series Switches*
- *clear ethernet-switching table*
- *show ethernet-switching mac-learning-log*

List of Sample Output

Output Fields For QFX Series, QFabric, NFX Series and EX4600:

The following table lists the output fields for the **show ethernet-switching table** command on QFX Series, QFabric, NFX Series and EX4600. Output fields are listed in the approximate order in which they appear.

Table 29: show ethernet-switching table Output Fields

Field Name	Field Description	Level of Output
VLAN	Name of a VLAN.	All levels
MAC address	MAC address associated with the VLAN.	All levels
Type	Type of MAC address: • **static**—The MAC address is manually created. • **learn**—The MAC address is learned dynamically from a packet's source MAC address. • **flood**—The MAC address is unknown and flooded to all members.	All levels
Age	Time remaining before the entry ages out and is removed from the Ethernet switching table.	All levels
Interfaces	Interface associated with learned MAC addresses or with the **All-members** option (flood entry).	All levels
Learned	For learned entries, the time at which the entry was added to the Ethernet switching table.	**detail**, **extensive**

For EX Series switches:

The following table lists the output fields for the **show ethernet-switching table** command on EX Series switches. Output fields are listed in the approximate order in which they appear.

Table 30: show ethernet-switching table Output Fields

Field Name	Field Description	Level of Output
VLAN	The name of a VLAN.	All levels
Tag	The VLAN ID tag name or number.	**extensive**
MAC or **MAC address**	The MAC address associated with the VLAN.	All levels
Type	The type of MAC address. Values are: • **static**—The MAC address is manually created. • **learn**—The MAC address is learned dynamically from a packet's source MAC address. • **flood**—The MAC address is unknown and flooded to all members. • **persistent**—The learned MAC addresses that will persist across restarts of the switch or interface-down events.	All levels except **persistent-mac**
Type	The type of MAC address. Values are: • **installed**—addresses that are in the Ethernet switching table. • **uninstalled**—addresses that could not be installed in the table or were uninstalled in an interface-down event and will be reinstalled in the table when the interface comes back up.	**persistent-mac**
Age	The time remaining before the entry ages out and is removed from the Ethernet switching table.	All levels
Interfaces	Interface associated with learned MAC addresses or **All-members** (flood entry).	All levels
Learned	For learned entries, the time which the entry was added to the Ethernet switching table.	**detail**, **extensive**
Nexthop index	The next-hop index number.	**detail**, **extensive**
persistent-mac	**installed** indicates MAC addresses that are in the Ethernet switching table and **uninstalled** indicates MAC addresses that could not be installed in the table or were uninstalled in an interface-down event (and will be reinstalled in the table when the interface comes back up).	

For EX Series, MX Series and QFX Series:

The table describes the output fields for the **show ethernet-switching table** command on EX Series, MX Series and QFX Series. Output fields are listed in the approximate order in which they appear.

Table 31: show ethernet-switching table Output fields

Field Name	Field Description
Routing instance	Name of the routing instance.
VLAN name	Name of the VLAN.

Table 31: show ethernet-switching table Output fields (continued)

Field Name	Field Description
MAC address	MAC address or addresses learned on a logical interface.
MAC flags	Status of MAC address learning properties for each interface: • **S**—Static MAC address is configured. • **D**—Dynamic MAC address is configured. • **L**—Locally learned MAC address is configured. • **SE**—MAC accounting is enabled. • **NM**—Non-configured MAC. • **R**—Locally learned MAC address is configured.
Age	This field is not supported.
Logical interface	Name of the logical interface.
Active source	IP address of remote entity on which MAC address is learned.
MAC count	Number of MAC addresses learned on the specific routing instance or interface.
Learning interface	Name of the logical interface on which the MAC address was learned.
Learning VLAN	VLAN ID of the routing instance or VLAN in which the MAC address was learned.
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning-tree-protocolepoch number identifying when the MAC address was learned. Used for debugging.
Sequence number	Sequence number assigned to this MAC address. Used for debugging.
Learning mask	Mask of the Packet Forwarding Engines where this MAC address was learned. Used for debugging.
IPC generation	Creation time of the logical interface when this MAC address was learned. Used for debugging.

For SRX Series:

Table 32 on page 1135 lists the output fields for the **show ethernet-switching table** command. Output fields are listed in the approximate order in which they appear.

Table 32: show ethernet-switching table Output Fields

Field Name	Field Description
VLAN	The name of a VLAN.

Table 32: show ethernet-switching table Output Fields (continued)

Field Name	Field Description
MAC address	The MAC address associated with the VLAN.
Type	The type of MAC address. Values are: • static—The MAC address is manually created. • learn—The MAC address is learned dynamically from a packet's source MAC address. • flood—The MAC address is unknown and flooded to all members.
Age	The time remaining before the entry ages out and is removed from the Ethernet switching table.
Interfaces	Interface associated with learned MAC addresses or All-members (flood entry).
Learned	For learned entries, the time which the entry was added to the Ethernet switching table.

Sample Output

show ethernet-switching table (Enhanced Layer 2 Software on QFX Series, QFabric, NFX Series and EX460)

```
user@switch> 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 : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    vlan1               b0:c6:9a:ca:3c:01   D             -    ae1.0

    vlan1               b0:c6:9a:ca:3c:03   D             -    ae1.0


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 : 2 entries, 2 learned

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    vlan10              b0:c6:9a:ca:3c:01   D             -    ae1.0

    vlan10              b0:c6:9a:ca:3c:03   D             -    ae1.0


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 : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    vlan2               b0:c6:9a:ca:3c:01   D             -    ae1.0

    vlan2               b0:c6:9a:ca:3c:03   D             -    ae1.0


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 : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    vlan3               b0:c6:9a:ca:3c:01   D             -    ae1.0

    vlan3               b0:c6:9a:ca:3c:03   D             -    ae1.0


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 : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    vlan4               b0:c6:9a:ca:3c:01   D             -    ae1.0

    vlan4               b0:c6:9a:ca:3c:03   D             -    ae1.0
```

show ethernet-switching table (QFX Series, QFabric, NFX Series and EX460)

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

Ethernet-switching table: 57 entries, 17 learned
  VLAN              MAC address       Type         Age Interfaces
  F2                *                 Flood          - All-members
  F2                00:00:05:00:00:03 Learn          0 xe-0/0/44.0
  F2                00:19:e2:50:7d:e0 Static         - Router
  Linux             *                 Flood          - All-members
  Linux             00:19:e2:50:7d:e0 Static         - Router
  Linux             00:30:48:90:54:89 Learn          0 xe-0/0/47.0
  T1                *                 Flood          - All-members
  T1                00:00:05:00:00:01 Learn          0 xe-0/0/46.0
  T1                00:00:5e:00:01:00 Static         - Router
```

```
  T1                00:19:e2:50:63:e0 Learn          0 xe-0/0/46.0
  T1                00:19:e2:50:7d:e0 Static         - Router
  T10               *                 Flood          - All-members
  T10               00:00:5e:00:01:09 Static         - Router
  T10               00:19:e2:50:63:e0 Learn          0 xe-0/0/46.0
  T10               00:19:e2:50:7d:e0 Static         - Router
  T111              *                 Flood          - All-members
  T111              00:19:e2:50:63:e0 Learn          0 xe-0/0/15.0
  T111              00:19:e2:50:7d:e0 Static         - Router
  T111              00:19:e2:50:ac:00 Learn          0 xe-0/0/15.0
  T2                *                 Flood          - All-members
  T2                00:00:5e:00:01:01 Static         - Router
  T2                00:19:e2:50:63:e0 Learn          0 xe-0/0/46.0
  T2                00:19:e2:50:7d:e0 Static         - Router
  T3                *                 Flood          - All-members
  T3                00:00:5e:00:01:02 Static         - Router
  T3                00:19:e2:50:63:e0 Learn          0 xe-0/0/46.0
  T3                00:19:e2:50:7d:e0 Static         - Router
  T4                *                 Flood          - All-members
  T4                00:00:5e:00:01:03 Static         - Router
  T4                00:19:e2:50:63:e0 Learn          0 xe-0/0/46.0
[output truncated]
```

show ethernet-switching table (Private VLANs on QFX Series, QFabric, NFX Series and EX460)

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

Ethernet-switching table: 10 entries, 3 learned
  VLAN              MAC address       Type         Age Interfaces
  pvlan             *                 Flood          - All-members
  pvlan             00:10:94:00:00:02 Replicated       - xe-0/0/28.0
  pvlan             00:10:94:00:00:35 Replicated       - xe-0/0/46.0
  pvlan             00:10:94:00:00:46 Replicated       - xe-0/0/4.0
  c2                *                 Flood          - All-members
  c2                00:10:94:00:00:02 Learn          0 xe-0/0/28.0
  c1                *                 Flood          - All-members
  c1                00:10:94:00:00:46 Learn          0 xe-0/0/4.0
  __pvlan_pvlan_xe-0/0/46.0__ *       Flood          - All-members
  __pvlan_pvlan_xe-0/0/46.0__ 00:10:94:00:00:35 Learn        0 xe-0/0/46.0
```

show ethernet-switching table (Junos Fusion Data Center with EVPN on QFX Series switches)

```
user@switch> 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 : 30 entries, 30 learned
Routing instance : default-switch
   Vlan               MAC                MAC      Logical               Active

   name               address            flags    interface             source

   v100               00:31:46:e8:f9:d6   D        vtep.32768
```

```
192.168.2.22
   v100                 7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v100                 7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v101                 00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v101                 7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v101                 7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v102                 00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v102                 7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v102                 7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v103                 00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v103                 7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v103                 7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v3001                00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v3001                28:c0:da:6a:9f:c2   DL       ae11.0
   v3001                7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v3001                7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v3002                00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v3002                7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v3002                7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v3003                00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v3003                28:c0:da:6a:9f:c2   DL       ae11.0
   v3003                7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v3003                7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v3004                00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v3004                7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v3004                7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
   v3005                00:31:46:e8:f9:d6   D        vtep.32768
192.168.2.22
   v3005                28:c0:da:6a:9f:c2   DL       ae11.0
   v3005                7c:e2:ca:e2:75:7c   D        vtep.32771
192.168.4.44
   v3005                7c:e2:ca:e4:05:9a   D        vtep.32770
192.168.3.33
```

show ethernet-switching table brief (QFX Series, QFabric, NFX Series and EX460)

```
user@switch> show ethernet-switching table brief

Ethernet-switching table: 57 entries, 17 learned
  VLAN              MAC address       Type          Age Interfaces
  F2                *                 Flood           - All-members
  F2                00:00:05:00:00:03 Learn           0 xe-0/0/44.0
  F2                00:19:e2:50:7d:e0 Static          - Router
  Linux             *                 Flood           - All-members
  Linux             00:19:e2:50:7d:e0 Static          - Router
  Linux             00:30:48:90:54:89 Learn           0 xe-0/0/47.0
  T1                *                 Flood           - All-members
  T1                00:00:05:00:00:01 Learn           0 xe-0/0/46.0
  T1                00:00:5e:00:01:00 Static          - Router
  T1                00:19:e2:50:63:e0 Learn           0 xe-0/0/46.0
  T1                00:19:e2:50:7d:e0 Static          - Router
  T10               *                 Flood           - All-members
  T10               00:00:5e:00:01:09 Static          - Router
  T10               00:19:e2:50:63:e0 Learn           0 xe-0/0/46.0
  T10               00:19:e2:50:7d:e0 Static          - Router
  T111              *                 Flood           - All-members
  T111              00:19:e2:50:63:e0 Learn           0 xe-0/0/15.0
  T111              00:19:e2:50:7d:e0 Static          - Router
  T111              00:19:e2:50:ac:00 Learn           0 xe-0/0/15.0
  T2                *                 Flood           - All-members
  T2                00:00:5e:00:01:01 Static          - Router
  T2                00:19:e2:50:63:e0 Learn           0 xe-0/0/46.0
  T2                00:19:e2:50:7d:e0 Static          - Router
  T3                *                 Flood           - All-members
  T3                00:00:5e:00:01:02 Static          - Router
  T3                00:19:e2:50:63:e0 Learn           0 xe-0/0/46.0
  T3                00:19:e2:50:7d:e0 Static          - Router
  T4                *                 Flood           - All-members
  T4                00:00:5e:00:01:03 Static          - Router
  T4                00:19:e2:50:63:e0 Learn           0 xe-0/0/46.0
[output truncated]
```

show ethernet-switching table detail (QFX Series, QFabric, NFX Series and EX460)

```
user@switch> show ethernet-switching table detail

Ethernet-switching table: 57 entries, 17 learned
  F2, *
    Interface(s): xe-0/0/44.0
    Type: Flood
    Nexthop index: 0

  F2, 00:00:05:00:00:03
    Interface(s): xe-0/0/44.0
    Type: Learn, Age: 0, Learned: 2:03:09
    Nexthop index: 0

  F2, 00:19:e2:50:7d:e0
    Interface(s): Router
    Type: Static
    Nexthop index: 0
```

```
Linux, *
  Interface(s): xe-0/0/47.0
  Type: Flood
  Nexthop index: 0

Linux, 00:19:e2:50:7d:e0
  Interface(s): Router
  Type: Static
  Nexthop index: 0

Linux, 00:30:48:90:54:89
  Interface(s): xe-0/0/47.0
  Type: Learn, Age: 0, Learned: 2:03:08
  Nexthop index: 0

T1, *
  Interface(s): xe-0/0/46.0
  Type: Flood
  Nexthop index: 0

T1, 00:00:05:00:00:01
  Interface(s): xe-0/0/46.0
  Type: Learn, Age: 0, Learned: 2:03:07
  Nexthop index: 0

T1, 00:00:5e:00:01:00
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T1, 00:19:e2:50:63:e0
  Interface(s): xe-0/0/46.0
  Type: Learn, Age: 0, Learned: 2:03:07
  Nexthop index: 0

T1, 00:19:e2:50:7d:e0
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T10, *
  Interface(s): xe-0/0/46.0
  Type: Flood
  Nexthop index: 0

T10, 00:00:5e:00:01:09
  Interface(s): Router
  Type: Static
  Nexthop index: 0

T10, 00:19:e2:50:63:e0
  Interface(s): xe-0/0/46.0
  Type: Learn, Age: 0, Learned: 2:03:08
  Nexthop index: 0

T10, 00:19:e2:50:7d:e0
  Interface(s): Router
  Type: Static
  Nexthop index: 0
```

```
  T111, *
    Interface(s): xe-0/0/15.0
    Type: Flood
    Nexthop index: 0
[output truncated]
```

show ethernet-switching table extensive (QFX Series, QFabric, NFX Series and EX460)

```
user@switch> show ethernet-switching table extensive

Ethernet-switching table: 57 entries, 17 learned
  F2, *
    Interface(s): xe-0/0/44.0
    Type: Flood
    Nexthop index: 0

  F2, 00:00:05:00:00:03
    Interface(s): xe-0/0/44.0
    Type: Learn, Age: 0, Learned: 2:03:09
    Nexthop index: 0

  F2, 00:19:e2:50:7d:e0
    Interface(s): Router
    Type: Static
    Nexthop index: 0

  Linux, *
    Interface(s): xe-0/0/47.0
    Type: Flood
    Nexthop index: 0

  Linux, 00:19:e2:50:7d:e0
    Interface(s): Router
    Type: Static
    Nexthop index: 0

  Linux, 00:30:48:90:54:89
    Interface(s): xe-0/0/47.0
    Type: Learn, Age: 0, Learned: 2:03:08
    Nexthop index: 0

  T1, *
    Interface(s): xe-0/0/46.0
    Type: Flood
    Nexthop index: 0

  T1, 00:00:05:00:00:01
    Interface(s): xe-0/0/46.0
    Type: Learn, Age: 0, Learned: 2:03:07
    Nexthop index: 0

  T1, 00:00:5e:00:01:00
    Interface(s): Router
    Type: Static
    Nexthop index: 0

  T1, 00:19:e2:50:63:e0
```

```
    Interface(s): xe-0/0/46.0
    Type: Learn, Age: 0, Learned: 2:03:07
    Nexthop index: 0

  T1, 00:19:e2:50:7d:e0
    Interface(s): Router
    Type: Static
    Nexthop index: 0

  T10, *
    Interface(s): xe-0/0/46.0
    Type: Flood
    Nexthop index: 0

  T10, 00:00:5e:00:01:09
    Interface(s): Router
    Type: Static
    Nexthop index: 0

  T10, 00:19:e2:50:63:e0
    Interface(s): xe-0/0/46.0
    Type: Learn, Age: 0, Learned: 2:03:08
    Nexthop index: 0

  T10, 00:19:e2:50:7d:e0
    Interface(s): Router
    Type: Static
    Nexthop index: 0

  T111, *
    Interface(s): xe-0/0/15.0
    Type: Flood
    Nexthop index: 0
[output truncated]
```

show ethernet-switching table interface (QFX Series, QFabric, NFX Series and EX460)

```
user@switch> show ethernet-switching table interface xe-0/0/1

Ethernet-switching table: 1 unicast entries
  VLAN              MAC address       Type         Age Interfaces
  V1                *                 Flood          - All-members
  V1                00:00:05:00:00:05 Learn          0 xe-0/0/1.0
```

show ethernet-switching table (EX Series switches)

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

Ethernet-switching table: 57 entries, 15 learned, 2 persistent
  VLAN              MAC address       Type         Age Interfaces
  F2                *                 Flood          - All-members
  F2                00:00:05:00:00:03 Learn          0 ge-0/0/44.0
  F2                00:19:e2:50:7d:e0 Static         - Router
  Linux             *                 Flood          - All-members
  Linux             00:19:e2:50:7d:e0 Static         - Router
  Linux             00:30:48:90:54:89 Learn          0 ge-0/0/47.0
```

```
  T1                *                 Flood          - All-members
  T1                00:00:05:00:00:01 Persistent     0 ge-0/0/46.0
  T1                00:00:5e:00:01:00 Static         - Router
  T1                00:19:e2:50:63:e0 Persistent     0 ge-0/0/46.0
  T1                00:19:e2:50:7d:e0 Static         - Router
  T10               *                 Flood          - All-members
  T10               00:00:5e:00:01:09 Static         - Router
  T10               00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
  T10               00:19:e2:50:7d:e0 Static         - Router
  T111              *                 Flood          - All-members
  T111              00:19:e2:50:63:e0 Learn          0 ge-0/0/15.0
  T111              00:19:e2:50:7d:e0 Static         - Router
  T111              00:19:e2:50:ac:00 Learn          0 ge-0/0/15.0
  T2                *                 Flood          - All-members
  T2                00:00:5e:00:01:01 Static         - Router
  T2                00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
  T2                00:19:e2:50:7d:e0 Static         - Router
  T3                *                 Flood          - All-members
  T3                00:00:5e:00:01:02 Static         - Router
  T3                00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
  T3                00:19:e2:50:7d:e0 Static         - Router
  T4                *                 Flood          - All-members
  T4                00:00:5e:00:01:03 Static         - Router
  T4                00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
[output truncated]
```

show ethernet-switching table brief (EX Series switches)

```
user@switch> show ethernet-switching table brief
```

```
Ethernet-switching table: 57 entries, 15 learned, 2 persistent entries
  VLAN              MAC address       Type          Age Interfaces
  F2                *                 Flood          - All-members
  F2                00:00:05:00:00:03 Learn          0 ge-0/0/44.0
  F2                00:19:e2:50:7d:e0 Static         - Router
  Linux             *                 Flood          - All-members
  Linux             00:19:e2:50:7d:e0 Static         - Router
  Linux             00:30:48:90:54:89 Learn          0 ge-0/0/47.0
  T1                *                 Flood          - All-members
  T1                00:00:05:00:00:01 Persistent     0 ge-0/0/46.0
  T1                00:00:5e:00:01:00 Static         - Router
  T1                00:19:e2:50:63:e0 Persistent     0 ge-0/0/46.0
  T1                00:19:e2:50:7d:e0 Static         - Router
  T10               *                 Flood          - All-members
  T10               00:00:5e:00:01:09 Static         - Router
  T10               00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
  T10               00:19:e2:50:7d:e0 Static         - Router
  T111              *                 Flood          - All-members
  T111              00:19:e2:50:63:e0 Learn          0 ge-0/0/15.0
  T111              00:19:e2:50:7d:e0 Static         - Router
  T111              00:19:e2:50:ac:00 Learn          0 ge-0/0/15.0
  T2                *                 Flood          - All-members
  T2                00:00:5e:00:01:01 Static         - Router
  T2                00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
  T2                00:19:e2:50:7d:e0 Static         - Router
  T3                *                 Flood          - All-members
  T3                00:00:5e:00:01:02 Static         - Router
  T3                00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
```

```
  T3                00:19:e2:50:7d:e0 Static         - Router
  T4                *                 Flood          - All-members
  T4                00:00:5e:00:01:03 Static         - Router
  T4                00:19:e2:50:63:e0 Learn          0 ge-0/0/46.0
[output truncated]
```

show ethernet-switching table detail (EX Series switches)

```
user@switch> show ethernet-switching table detail
```

```
Ethernet-switching table: 5 entries, 2 learned entries
 VLAN: default, Tag: 0, MAC: *, Interface: All-members
  Interfaces:
   ge-0/0/11.0, ge-0/0/20.0, ge-0/0/30.0, ge-0/0/36.0, ge-0/0/3.0
  Type: Flood
  Nexthop index: 1307

 VLAN: default, Tag: 0, MAC: 00:1f:12:30:b8:83, Interface: ge-0/0/3.0
   Type: Learn, Age: 0, Learned: 20:09:26
   Nexthop index: 1315

 VLAN: v1, Tag: 101, MAC: *, Interface: All-members
  Interfaces:
   ge-0/0/31.0
  Type: Flood
  Nexthop index: 1313

 VLAN: v1, Tag: 101, MAC: 00:1f:12:30:b8:89, Interface: ge-0/0/31.0
  Type: Learn, Age: 0, Learned: 20:09:25
  Nexthop index: 1312

 VLAN: v2, Tag: 102, MAC: *, Interface: All-members
  Interfaces:
   ae0.0
  Type: Flood
  Nexthop index: 1317
```

show ethernet-switching table extensive (EX Series switches)

```
user@switch> show ethernet-switching table extensive
```

```
Ethernet-switching table: 3 entries, 1 learned, 5 persistent entries

    VLAN: v1, Tag: 10, MAC: *, Interface: All-members
    Interfaces:
                ge-0/0/14.0, ge-0/0/1.0, ge-0/0/2.0, ge-0/0/3.0, ge-0/0/4.0,
                ge-0/0/5.0, ge-0/0/6.0, ge-0/0/7.0, ge-0/0/8.0, ge-0/0/10.0,
                ge-0/0/0.0
    Type: Flood
    Nexthop index: 567

    VLAN: v1, Tag: 10, MAC: 00:21:59:c6:93:22, Interface: Router
    Type: Static
    Nexthop index: 0

    VLAN: v1, Tag: 10, MAC: 00:21:59:c9:9a:4e, Interface: ge-0/0/14.0
```

```
   Type: Learn, Age: 0, Learned: 18:40:50
   Nexthop index: 564
```

show ethernet-switching table persistent-mac (EX Series switches)

```
user@switch> show ethernet-switching table persistent-mac

VLAN              MAC address       Type       Interface
default           00:10:94:00:00:02 installed    ge-0/0/42.0
default           00:10:94:00:00:03 installed    ge-0/0/42.0
default           00:10:94:00:00:04 installed    ge-0/0/42.0
default           00:10:94:00:00:05 installed    ge-0/0/42.0
default           00:10:94:00:00:06 installed    ge-0/0/42.0
default           00:10:94:00:05:02 uninstalled ge-0/0/16.0
default           00:10:94:00:06:03 uninstalled ge-0/0/16.0
default           00:10:94:00:07:04 uninstalled ge-0/0/16.0
```

show ethernet-switching table persistent-mac interface ge-0/0/16.0 (EX Series switches)

```
VLAN              MAC address       Type       Interface
default           00:10:94:00:05:02 uninstalled ge-0/0/16.0
default           00:10:94:00:06:03 uninstalled ge-0/0/16.0
default           00:10:94:00:07:04 uninstalled ge-0/0/16.0
```

show ethernet-switching table (EX Series, MX Series and QFX Series)

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

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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN101             88:e0:f3:bb:07:f0   D             -    ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN102             88:e0:f3:bb:07:f0   D             -    ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN103             88:e0:f3:bb:07:f0   D             -    ae20.0
```

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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN104             88:e0:f3:bb:07:f0   D             -    ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1101            00:1f:12:32:f5:c1   D             -    ae0.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1102            00:1f:12:32:f5:c1   D             -    ae0.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1103            00:1f:12:32:f5:c1   D             -    ae0.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1104            00:1f:12:32:f5:c1   D             -    ae0.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1105            00:1f:12:32:f5:c1   D             -    ae0.0


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

Routing instance : default-switch
```

```
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags               interface
    VLAN1106            00:1f:12:32:f5:c1   D             -     ae0.0
[...output truncated...]
```

show ethernet-switching table brief

```
user@host> show ethernet-switching table brief

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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags               interface
    VLAN101             88:e0:f3:bb:07:f0   D             -     ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags               interface
    VLAN102             88:e0:f3:bb:07:f0   D             -     ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags               interface
    VLAN103             88:e0:f3:bb:07:f0   D             -     ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags               interface
    VLAN104             88:e0:f3:bb:07:f0   D             -     ae20.0


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

Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags               interface
    VLAN1101            00:1f:12:32:f5:c1   D             -     ae0.0
[...output truncated...]
```

show ethernet-switching table count

```
user@host> show ethernet-switching table count
```

```
0 MAC address learned in routing instance default-switch VLAN VLAN1000
ae26.0:1000

1 MAC address learned in routing instance default-switch VLAN VLAN101
ae20.0:101

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID      MAC count      Static MAC count
           101              1                0

1 MAC address learned in routing instance default-switch VLAN VLAN102
ae20.0:102

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID      MAC count      Static MAC count
           102              1                0

1 MAC address learned in routing instance default-switch VLAN VLAN103
ae20.0:103

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID      MAC count      Static MAC count
           103              1                0

1 MAC address learned in routing instance default-switch VLAN VLAN104
ae20.0:104

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID      MAC count      Static MAC count
           104              1                0

0 MAC address learned in routing instance default-switch VLAN VLAN105
ae20.0:105

0 MAC address learned in routing instance default-switch VLAN VLAN106
ae20.0:106

0 MAC address learned in routing instance default-switch VLAN VLAN107
ae20.0:107

0 MAC address learned in routing instance default-switch VLAN VLAN108
ae20.0:108

0 MAC address learned in routing instance default-switch VLAN VLAN109
ae20.0:109

0 MAC address learned in routing instance default-switch VLAN VLAN110
ae20.0:110

1 MAC address learned in routing instance default-switch VLAN VLAN1101
ae0.0:1101

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID      MAC count      Static MAC count
          1101              1                0

1 MAC address learned in routing instance default-switch VLAN VLAN1102
ae0.0:1102
```

```
  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID      MAC count      Static MAC count
          1102              1                 0
[...output truncated...]
```

show ethernet-switching table extensive

user@host> **show ethernet-switching table extensive**

```
MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 101
   Learning interface: ae20.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                            Sequence number: 2
   Learning mask: 0x00000008

MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 102
   Learning interface: ae20.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                            Sequence number: 2
   Learning mask: 0x00000008

MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 103
   Learning interface: ae20.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                            Sequence number: 2
   Learning mask: 0x00000008

MAC address: 88:e0:f3:bb:07:f0
  Routing instance: default-switch
VLAN ID: 104
   Learning interface: ae20.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                            Sequence number: 2
   Learning mask: 0x00000008

MAC address: 00:1f:12:32:f5:c1
  Routing instance: default-switch
VLAN ID: 1101
   Learning interface: ae0.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                            Sequence number: 2
   Learning mask: 0x00000008

MAC address: 00:1f:12:32:f5:c1
  Routing instance: default-switch
VLAN ID: 1102
   Learning interface: ae0.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                            Sequence number: 2
   Learning mask: 0x00000008
```

```
MAC address: 00:1f:12:32:f5:c1
  Routing instance: default-switch
VLAN ID: 1103
   Learning interface: ae0.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                          Sequence number: 2
   Learning mask: 0x00000008

MAC address: 00:1f:12:32:f5:c1
  Routing instance: default-switch
VLAN ID: 1104
   Learning interface: ae0.0
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                          Sequence number: 2
   Learning mask: 0x00000008
```

Sample Output

show ethernet-switching table detail (SRX Series)

```
user@host> show ethernet-switching table detail

Ethernet-switching table: 57 entries, 17 learned
F2, *
Interface(s): ge-0/0/44.0
Type: Flood
F2, 00:00:5E:00:53:AC
Interface(s): ge-0/0/44.0
Type: Learn, Age: 0, Learned: 2:03:09
F2, 00:00:5E:00:53:AA
Interface(s): Router
Type: Static
Linux, *
Interface(s): ge-0/0/47.0
Type: Flood
Linux, 00:00:5E:00:53:AB
Interface(s): Router
Type: Static
Linux, 00:00:5E:00:53:AC
Interface(s): ge-0/0/47.0
Type: Learn, Age: 0, Learned: 2:03:08
T1, *
Interface(s): ge-0/0/46.0
Type: Flood
T1, 00:00:5E:00:53:AD
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AE
Interface(s): Router
Type: Static
T1, 00:00:5E:00:53:AF
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AG
Interface(s): Router
Type: Static
T10, *
Interface(s): ge-0/0/46.0
```

```
Type: Flood
T10, 00:00:5E:00:53:AH
Interface(s): Router
Type: Static
T10, 00:00:5E:00:53:AI
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:08
T10, 00:00:5E:00:53:AJ
Interface(s): Router
Type: Static
T111, *
Interface(s): ge-0/0/15.0
Type: Flood
[output truncated]
```

Sample Output

show ethernet-switching table extensive (SRX Series)

user@host> **show ethernet-switching table extensive**

```
Ethernet-switching table: 57 entries, 17 learned
F2, *
Interface(s): ge-0/0/44.0
Type: Flood
F2, 00:00:5E:00:53:AC
Interface(s): ge-0/0/44.0
Type: Learn, Age: 0, Learned: 2:03:09
F2, 00:00:5E:00:53:AA
Interface(s): Router
Type: Static
Linux, *
Interface(s): ge-0/0/47.0
Type: Flood
Linux, 00:00:5E:00:53:AB
Interface(s): Router
Type: Static
Linux, 00:00:5E:00:53:AC
Interface(s): ge-0/0/47.0
Type: Learn, Age: 0, Learned: 2:03:08
T1, *
Interface(s): ge-0/0/46.0
Type: Flood
T1, 00:00:5E:00:53:AD
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AE
Interface(s): Router
Type: Static
T1, 00:00:5E:00:53:AF
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:07
T1, 00:00:5E:00:53:AG
Interface(s): Router
Type: Static
T10, *
Interface(s): ge-0/0/46.0
Type: Flood
```

```
T10, 00:00:5E:00:53:AH
Interface(s): Router
Type: Static
T10, 00:00:5E:00:53:AI
Interface(s): ge-0/0/46.0
Type: Learn, Age: 0, Learned: 2:03:08
T10, 00:00:5E:00:53:AJ
Interface(s): Router
Type: Static
T111, *
Interface(s): ge-0/0/15.0
Type: Flood
[output truncated]
```

Sample Output

show ethernet-switching table interface ge-0/0/1 (SRX Series)

```
user@host> show ethernet-switching table interface ge-0/0/1

Ethernet-switching table: 1 unicast entries
VLAN     MAC address         Type     Age Interfaces
V1       *                   Flood    - All-members
V1       00:00:5E:00:53:AF   Learn    0 ge-0/0/1.0
```

show evpn arp-table

Syntax

```
show evpn arp-table
<address>
<brief |count | detail | extensive>
<instance instance-name>
<logical-system logical-system-name>
```

Release Information

Command introduced in Junos OS Release 15.1 for EX Series switches.
logical system option introduced in Junos OS Release 18.1R1.

Description

Show Ethernet VPN (EVPN) Address Resolution Protocol (ARP) entries associated with learned MAC addresses.

Options

none—Display brief information about the EVPN ARP table.

address—(Optional) Display ARP information for the specified MAC address.

brief | count | detail |extensive—(Optional) Display the specified level of output.

instance <*instance-name*>—oot(Optional) Display ARP information for the specified routing instance .

logical-system <*logical-system-name*>—(Optional) Display ARP information for the specified logical system or all logical systems.

Required Privilege Level

view

List of Sample Output

show evpn arp-table on page 1155
show evpn arp-table 00:05:86:a0:dc:f0 (MAC address) on page 1155
show evpn arp-table brief on page 1155
show evpn arp-table detail on page 1155
show evpn arp-table count on page 1156
show evpn arp-table extensive on page 1156
show evpn arp-table instance evpn1 on page 1156

Output Fields

Table 33 on page 1154 lists the output fields for the **show evpn arp-table** command. Output fields are listed in the approximate order in which they appear.

Table 33: show evpn arp-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the ARP table.	All levels

Table 33: show evpn arp-table Output Fields (continued)

Field Name	Field Description	Level of Output
MAC address	MAC addresses learned through ARP.	**brief**, **detail**, **extensive**, **instance**, **mac-address**,,
Logical Interface	Logical interface associated with the routing instance in which the ARP INET address is learned.	**brief**, **instance**, **mac-address**,,
Routing instance	Routing instance in which the ARP INET address is learned.	all levels
Bridging domain	Bridging domain in which the ARP INET address is learned.	all levels
Learning interface	Interface on which the ARP INET address is learned.	**detail**, **extensive**
Count	Indicates the number of ARP INET addresses learned in a routing instance in a bridge domain.	**count**

Sample Output

show evpn arp-table

```
user@host> show evpn arp-table

INET              MAC               Logical      Routing      Bridging
address           address           interface    instance     domain
203.0.113.2       00:05:86:a0:dc:f0 irb.0        evpn1        __evpn1__
```

show evpn arp-table 00:05:86:a0:dc:f0 (MAC address)

```
user@host> show evpn arp-table 00:05:86:a0:dc:f0

INET              MAC               Logical      Routing      Bridging
address           address           interface    instance     domain
203.0.113.2       00:05:86:a0:dc:f0 irb.0        evpn1        __evpn1__
```

show evpn arp-table brief

```
user@host> show evpn arp-table brief

INET              MAC               Logical      Routing      Bridging
address           address           interface    instance     domain
203.0.113.2       00:05:86:a0:dc:f0 irb.0        evpn1        __evpn1__
```

show evpn arp-table detail

```
user@host> show evpn arp-table detail
```

```
INET address: 203.0.113.2
MAC address: 00:05:86:a0:dc:f0
  Routing instance: evpn1
   Bridging domain: __evpn1__
   Learning interface: irb.0
```

show evpn arp-table count

```
user@switch> show evpn arp-table count
```

```
1 ARP INET addresses learned in routing instance evpn1 bridge domain __evpn1__
```

show evpn arp-table extensive

```
user@host> show evpn arp-table extensive
```

```
INET address: 203.0.113.2
MAC address: 00:05:86:a0:dc:f0
  Routing instance: evpn1
   Bridging domain: __evpn1__
   Learning interface: irb.0
```

show evpn arp-table instance evpn1

```
user@host> show evpn arp-table instance evpn1
```

```
INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
203.0.113.2           00:05:86:a0:dc:f0  irb.0         evpn1        __evpn1__
```

show evpn database

Syntax

```
show evpn database
<extensive>
<instance instance-name>
<interface interface-name>
<logical-system logical-system-name>
<mac-address address>
<neighbor neighbor-name>
<origin origin-name>
<state state-name>
<vlan-id vlan-id>
```

Release Information

Command introduced in Junos OS Release 14.2 for EX Series switches.
Command introduced in Junos OS Release 16.1 for MX Series routers.
Command introduced in Junos OS Release 17.1 on MX Series routers with MPCs.

Description

Show Ethernet VPN (EVPN) database information.

When up MEP is configured,, the **show evpn instance** displays the CFM MAC local interface with a naming convention as .local..*number* (for example, .local..8).

Options

none—Display brief information about the EVPN database.

extensive —(Optional) Display detailed information about the EVPN database.

instance ***instance-name***—(Optional) Display MAC addresses from the specified routing instance.

interface ***interface-name***—(Optional) Display the MAC address learned from the specified interface.

logical-system <***logical-system-name***>—(Optional) Display database information for the specified logical system or all logical systems.

mac-address ***address***—(Optional) Display the specified MAC address.

neighbor ***neighbor-name***—(Optional) Display the MAC address learned from the specified neighbor.

origin ***origin-name***—(Optional) Display the MAC address with the specified origin.

state ***state-name***—(Optional) Display the MAC address with the specified state.

vlan-id ***vlan-id***—(Optional) Display the MAC address with the specified VLAN.

Required Privilege Level

view

Related Documentation

- Example: Configuring an EVPN with IRB Solution on EX9200 Switches on page 636

List of Sample Output

Output Fields

Table 34 on page 1158 lists the output fields for the **show evpn database** command. Output fields are listed in the approximate order in which they appear.

Table 34: show evpn database Output Fields

Field Name	Field Description	Level of Output
Instance	Name of the routing instances.	All Levels
VNI	VXLAN network identifier.	All Levels
MAC address	MAC address of the routing instance.	All Levels
Origin	Specific origin of the MAC address.	All Levels
Timestamp	Month, day and time of generated command output.	All Levels
IP address	IP address.	All Levels
Prefix	Prefix address of the VXLAN network identifier.	**extensive**
Source	Source address of the VXLAN network identifier.	**extensive**
Rank	Level number of the VXLAN network identifier.	**extensive**
Status	Status of the of the VXLAN network identifier.	**extensive**
Remote origin	Specific remote origin of the VXLAN network identifier.	**extensive**
Mobility History	Information on a MAC address mobility history: • Mobility event time—Date and time of the MAC mobility event • Type—Origin of the learned MAC address (Local or Remote). • Source—Source of the learned MAC address (Local Interface, Remote PE IP address, or ESI) • Seq num—Sequence number associated with the mobility event	**extensive**
Mac label	Label advertised by the remote PE device when forwarding unicast traffic. In EVPN-MPLS, it is the MPLS label and in EVPN-VXLAN, it is the VNI .	**extensive**

Table 34: show evpn database Output Fields (continued)

Field Name	Field Description	Level of Output
Mobility Sequence number	Current sequence number associated with the MAC address. The minimum origin address is the IP address of the PE device with the lowest IP address that is sending a MAC advertisement.	**extensive**
State	Information collected in the EVPN database from different flag states including Duplicate-Detected, Local-Pinned, and Remote-Pinned.	**extensive**
IP address	IPv4 or IPv6 address for the MAC address.	**extensive**
L3 route	Route installed on the EVPN IRB interface.	**extensive**
L3 context	Name of the routing instance that has the Layer 3 routes installed for an EVPN IRB interface, typically a virtual routing and forward (VRF) routing instance.	**extensive**

Sample Output

show evpn database

```
user@host> show evpn database

Instance: bd_red
VNI  MAC address        Origin             Timestamp        IP address
50    1a:1b:1b:1b:1b:1b  192.0.2.3          Jun 12 21:57:17
50    1a:1c:1c:1c:1c:1c  ge-2/3/0.0         Jun 12 22:05:37
50    b0:c6:9a:e9:cd:d8  192.0.2.3          Jun 12 21:57:17
50    b0:c6:9a:ea:87:42  ge-2/3/0.0         Jun 12 22:05:38

Instance: evpn-0
VNI   MAC address        Origin           Timestamp        IP address
4000  1a:1a:1a:1a:01:01  ge-2/3/0.10      Jun 12 21:53:01
4000  1a:1a:1a:1a:01:02  ge-2/3/0.10      Jun 12 21:53:01
```

show evpn database extensive

```
user@host> show evpn database extensive

Instance: ALPHA

VN Identifier: 9100, Prefix: 10.0.0.0/24
  Source: 10.255.0.2, Rank: 1, Status: Active
    Timestamp: May 22 17:16:12 (0x555fc6cc)

VN Identifier: 9100, Prefix: 198.51.100.0/24
  Source: 05:00:00:00:64:00:00:23:8c:00, Rank: 1, Status: Active
    Remote origin: 10.255.0.2
    Remote origin: 10.255.0.3
    Timestamp: May 22 17:14:20 (0x555fc65c)
```

```
VN Identifier: 9100, Prefix: 192.0.2.0/24
  Source: irb.0, Rank: 1, Status: Active
    Timestamp: May 22 17:16:12 (0x555fc6cc)

VN Identifier: 9100, Prefix: 203.0.113.0/24
  Source: 10.255.0.3, Rank: 1, Status: Active
    Timestamp: May 22 17:16:12 (0x555fc6cc)
```

show evpn database extensive instance

```
user@PE1> show evpn database instance ALPHA mac-address 00:00:00:00:00:01 extensive

Instance: ALPHA

VLAN ID: 100, MAC address: 00:00:00:00:00:01
  Nexthop ID: 1048575
  Mobility history
    Mobility event time     Type    Source                                  Seq
 num
    Jul 10 16:26:14.920136  Remote  10.255.0.3                               13
    Jul 10 16:26:16.174769  Local   ge-0/0/2.0                               14
    Jul 10 16:26:17.868187  Remote  10.255.0.3                               15
    Jul 10 16:26:19.129879  Local   ge-0/0/2.0                               16
    Jul 10 16:26:25.972747  Remote  10.255.0.3                               17
  Source: 10.255.0.3, Rank: 1, Status: Active
    MAC label: 299776
    Mobility sequence number: 17 (minimum origin address 10.255.0.3)
    Timestamp: Jul 10 16:26:25 (0x59640d21)
    State: <Remote-To-Local-Adv-Done>
    IP address: 10.0.0.3
      L3 route: 10.0.0.3/32, L3 context: DELTA (irb.0)
```

show evpn database extensive (Duplicate MAC Address)

```
user@PE1> show evpn database instance ALPHA mac-address 00:00:00:00:00:012extensive

Instance: ALPHA

VLAN ID: 100, MAC address: 00:00:00:00:00:02
  State: 0x1 <Duplicate-Detected>
  Mobility history
    Mobility event time     Type    Source                                  Seq
 num
    Aug 03 17:22:28.585619  Local   ge-0/0/2.0                               31
    Aug 03 17:22:30.307198  Remote  10.255.0.3                               32
    Aug 03 17:22:37.611786  Local   ge-0/0/2.0                               33
    Aug 03 17:22:39.289357  Remote  10.255.0.3                               34
    Aug 03 17:22:45.609449  Local   ge-0/0/2.0                               35
  Source: ge-0/0/2.0, Rank: 1, Status: Active
    Mobility sequence number: 35 (minimum origin address 10.255.0.2)
    Timestamp: Aug 03 17:22:44 (0x5983be54)
    State: <Local-MAC-Only Local-To-Remote-Adv-Allowed>
    MAC advertisement route status: Not created (duplicate MAC suppression)
    IP address: 10.0.0.2
  Source: 10.255.0.3, Rank: 2, Status: Inactive
    MAC label: 300176
```

```
Mobility sequence number: 34 (minimum origin address 10.255.0.3)
Timestamp: Aug 03 17:22:39 (0x5983be4f)
State: <>
MAC advertisement route status: Not created (inactive source)
IP address: 10.0.0.3
```

show evpn flood

Syntax

```
show evpn flood
event-queue
<instance instance-name>
<logical-system logical-system-name>
<route route-name>
```

Release Information Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Show Ethernet VPN (EVPN) flooding information.

Options

none—Display brief information about EVPN flooding.

brief | detail |extensive | summary—(Optional) Display the specified level of output.

event-queue—(Optional) Display the queue of pending EVPN flood events.

instance *instance-name*—(Optional) Display flooding information for the specified routing instance.

logical-system <*logical-system-name*>—(Optional) Display flooding information for the specified logical system or all logical systems.

route *route-name*—(Optional) Display flooding information for the specified route.

Required Privilege Level view

Related Documentation

- Example: Configuring an EVPN with IRB Solution on EX9200 Switches on page 636

show evpn instance

Syntax

```
show evpn instance
<brief | extensive>
<backup-forwarder>
<designated-forwarder>
<esi esi>
<instance-name>
<neighbor neighbor-address>
```

Release Information

Command introduced in Junos OS Release 14.1 for MX series Routers.
Command introduced in Junos OS Release 14.2 for EX Series switches.
Command introduced in Junos OS Release 17.4R1 for QFX10000 switches.

Description

Show Ethernet VPN (EVPN) routing instance information.

When up MEP is configured for an EVPN instance (EVI), the **show evpn instance** displays a default interface without any configuration with a naming convention as .local..*number* (for example, .local..8).

Options

none—Display brief information about the EVPN routing instance.

brief | extensive—(Optional) Display the specified level of output.

backup-forwarder—(Optional) Display IP addresses of all the backup designated forwarders for the Ethernet segment.

designated-forwarder—(Optional) Display the IP address of the designated forwarder for the Ethernet segment.

esi *esi*—(Optional) Display brief information about the routing instance associated with the specified Ethernet segment identifier (ESI) value.

instance-name—(Optional) Display information about the specified routing instance.

neighbor *neighbor-address*—(Optional) Display the IP address of the EVPN neighbor.

Required Privilege Level

view

List of Sample Output

show evpn instance esi designated-forwarder (Instance Name with Ethernet Segment Identifier) on page 1172
show evpn instance (CFM up) on page 1172

Output Fields Table 35 on page 1164 lists the output fields for the **show evpn instance** command. Output fields are listed in the approximate order in which they appear.

Table 35: show evpn instance Output Fields

Field Name	Field Description	Level of Output
Instance	Names of the routing instances.	All levels
Intfs	Total number of interfaces participating in each routing instance, and number of interfaces that are up.	**brief**
IRB intfs	Statistics on the number of integrated routing and bridging (IRB) interfaces for each routing instance: • **Total**—Total number of IRB interfaces. • **Up**—Number of active IRB interfaces. • **Nbrs**—Number of neighbor IRB interfaces.	**brief**
MH ESIs	Number of Ethernet segments per routing instance that connect to a multihomed customer site.	**brief**
MAC addresses	Number of local and remote MAC addresses for each routing instance.	**brief**
Route Distinguisher	Unique route distinguisher associated with this routing instance.	none
VLAN ID	VLAN identifier.	none
Label allocation mode	Label allocation policy for the routing instance.	none
Encapsulation type	Encapsulation type for VXLAN EVPN instances (EVIs).	**extensive**
Per-instance MAC route label	Label of MAC route for each routing instance.	none
Duplicate MAC detection threshold	Number of MAC mobility events detected for a given MAC address before it is identified as a duplicate MAC address.	**extensive**
Duplicate MAC detection window	The time interval used in detecting a duplicate MAC address.	**extensive**
Duplicate MAC auto-recovery time	Length of time a device suppresses a duplicate MAC address. At the end of this duration, MAC address updates will resume.	**extensive**

Table 35: show evpn instance Output Fields (continued)

Field Name	Field Description	Level of Output
DF Election preference	Preference value used for the designated forwarder (DF) election: • Highest preference—Default DF election preference. • Lowest preference—Based on the configuration of the **designated-forwarder-preference-least** statement at the **[edit routing-instance *routing-instance-name* protocols evpn]** hierarchy level.	**extensive**
Per-instance multicast route label	Label of the multicast route for each routing instance.	none
Total MAC addresses	Total number of local and remote MAC addresses received for each routing instance.	**extensive**
Default gateway MAC addresses	Number of local and remote MAC addresses serving as a default gateway in this routing instance.	**extensive**
Number of local interfaces	Number of local interfaces belonging to this routing instance.	**extensive**
Number of local interfaces up	Number of active local interfaces belonging to this routing instance.	**extensive**
Interface name	Name of interfaces that belong to this routing instance.	**extensive**
ESI	Ethernet segment identifier (ESI) value of the interfaces belonging to this routing instance.	**extensive**
DF Election Algorithm	DF election type: • MOD based—Default DF election algorithm based on the modulo operation. • Preference based—DF election based on the configured preference values for an ESI.	**extensive**
Preference	Preference value for EVPN Multihoming DF election.	**extensive**
Mode	Mode of operation for each routing instance: • **single-homed**—Default mode and does not require Ethernet segment values to be configured. • **single-active**—EVPN active-standby multihoming mode of operation.	**extensive**
SH label	Split horizon label used for the active-standby multihoming mode of operation.	**extensive**
Number of IRB interfaces	Number of IRB interfaces that belong to this routing instance.	**extensive**

Table 35: show evpn instance Output Fields (continued)

Field Name	Field Description	Level of Output
Number of protect interfaces	Number of protect interfaces that belong to this routing instance.	**extensive**
Inteface name	Name of the primary interface.	**extensive**
Protect Interface name	Name of the protect interface.	**extensive**
Status	Protected status of the primary interface. The status is either Protect-inactive or Protect-active.	**extensive**
L3 context	Names of routing instances that have the Layer 3 routes installed for an EVPN IRB interface, typically a virtual routing and forwarding (VRF) routing instance.	**extensive**
Number of neighbors	Number of neighbors connected to this routing instance and their IP addresses.	**extensive**
MAC address advertisement	Number of MAC address advertisements received from the neighbor.	**extensive**
MAC+IP address advertisement	Number of MAC and IP address advertisements received from the neighbor.	**extensive**
Inclusive multicast	Number of inclusive multicast routes received from the neighbor.	**extensive**
Ethernet auto-discovery	Number of autodiscovery routes per Ethernet segment received from the neighbor.	**extensive**
Number of ethernet segments	Total number of Ethernet segments for the routing instance.	**extensive**
Designated forwarder	IP address of the designated forwarder (DF) for the Ethernet segment.	**extensive**
Backup forwarder	IP address of all the backup designated forwarders or routers that are not designated forwarders for the Ethernet segment. NOTE: Immediately after an EVPN interface Ethernet segment identifier value is changed and the new configuration is committed, the **Designated forwarder** information changes to **DF not elected yet** and the backup forwarder information is not displayed until after the election is complete.	**extensive**

Sample Output

show evpn instance brief

```
user@host> show evpn instance brief
```

```
                        Intfs       IRB intfs         MH      MAC addresses
Instance                Total   Up  Total   Up  Nbrs  ESIs    Local  Remote
ALPHA                     2     2      1    1     2     1        3       4
BETA                      2     2      1    1     2     1        2       4
__default_evpn__          0     0      0    0     1     0        0       0
```

show evpn instance

```
user@host> show evpn instance
```

```
Instance: black
  Route Distinguisher: 101:101
  VLAN ID: 100
  Label allocation mode: Per-instance
  Per-instance MAC route label: 299776
  Per-instance multicast route label: 299792
  Number of local interfaces: 1
  Number of local interfaces up: 1
    Interface name    Static MACs    ESI
    cbp-0.0                 0    0
  Number of neighbors: 1
    192.0.2.1
      Received routes
        MAC address advertisement:          1
        Ethernet auto-discovery:            0
        Inclusive multicast:                1
```

show evpn instance extensive (In Junos OS Release 16.1 and earlier)

```
user@host> show evpn instance extensive
```

```
Instance: ALPHA
  Route Distinguisher: 10.255.0.1:100
  Encapsulation type: VXLAN
  Per-instance MAC route label: 300144
  Per-instance multicast route label: 300160
  MAC database status                Local  Remote
    Total MAC addresses:                 3       4
    Default gateway MAC addresses:       1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                             Mode             SH label
    ae0.0           00:11:22:33:44:55:66:77:88:99   single-active
    ge-0/0/2.0      00:00:00:00:00:00:00:00:00:00   single-homed
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.0           DELTA
  Number of neighbors: 2
    10.255.0.2
      Received routes
        MAC address advertisement:             2
        MAC+IP address advertisement:          3
        Inclusive multicast:                   1
        Ethernet auto-discovery:               1
    10.255.0.3
      Received routes
        MAC address advertisement:             2
```

```
      MAC+IP address advertisement:           2
      Inclusive multicast:                    1
      Ethernet auto-discovery:                0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: BETA
  Route Distinguisher: 10.255.0.1:300
  Encapsulation type: VXLAN
  VLAN ID: 300
  Per-instance MAC route label: 300176
  Per-instance multicast route label: 300192
  MAC database status               Local  Remote
    Total MAC addresses:                3       4
    Default gateway MAC addresses:      1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                             Mode             SH label
    ae1.0           00:00:00:00:00:00:00:00:00:00   single-homed
    ge-0/0/4.0      00:22:44:66:88:00:22:44:66:88   single-active
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.1           DELTA
  Number of neighbors: 2
    10.255.0.2
      Received routes
        MAC address advertisement:            2
        MAC+IP address advertisement:         3
        Inclusive multicast:                  1
        Ethernet auto-discovery:              1
    10.255.0.3
      Received routes
        MAC address advertisement:            2
        MAC+IP address advertisement:         2
        Inclusive multicast:                  1
        Ethernet auto-discovery:              0
  Number of ethernet segments: 1
    ESI: 00:22:44:66:88:00:22:44:66:88
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: __default_evpn__
  Route Distinguisher: 10.255.0.1:0
  Encapsulation type: VXLAN
  VLAN ID: 0
  Per-instance MAC route label: 300208
  Per-instance multicast route label: 300224
  MAC database status               Local  Remote
    Total MAC addresses:                0       0
    Default gateway MAC addresses:      0       0
  Number of local interfaces: 0 (0 up)
  Number of IRB interfaces: 0 (0 up)
  Number of neighbors: 1
    10.255.0.2
      Received routes
        Ethernet auto-discovery:              0
        Ethernet Segment:                     2
  Number of ethernet segments: 0
```

show evpn instance extensive (In Junos OS Release 16.2 and later)

```
user@host> show evpn instance extensive
```

```
user@host> show evpn instance extensive

Instance: ALPHA
  Route Distinguisher: 10.255.0.1:100
  Encapsulation type: VXLAN
  Per-instance MAC route label: 300144
  Per-instance multicast route label: 300160
  MAC database status                 Local  Remote
    Total MAC addresses:                  3       4
    Default gateway MAC addresses:        1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                            Mode             SH label
    ae0.0           00:11:22:33:44:55:66:77:88:99  single-active
    ge-0/0/2.0      00:00:00:00:00:00:00:00:00:00  single-homed
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.0           DELTA
  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES
    10.255.0.2              2         3         1         1         0
    10.255.0.3              2         2         0         1         0
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: BETA
  Route Distinguisher: 10.255.0.1:300
  Encapsulation type: VXLAN
  VLAN ID: 300
  Per-instance MAC route label: 300176
  Per-instance multicast route label: 300192
  MAC database status                 Local  Remote
    Total MAC addresses:                  3       4
    Default gateway MAC addresses:        1       2
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                            Mode             SH label
    ae1.0           00:00:00:00:00:00:00:00:00:00  single-homed
    ge-0/0/4.0      00:22:44:66:88:00:22:44:66:88  single-active
  Number of IRB interfaces: 1 (1 up)
    Interface name  L3 context
    irb.1           DELTA
  Number of neighbors: 2
    Address               MAC    MAC+IP        AD        IM        ES
    10.255.0.2              2         3         1         1         0
    10.255.0.3              2         2         0         1         0
  Number of ethernet segments: 1
    ESI: 00:22:44:66:88:00:22:44:66:88
      Designated forwarder: 10.255.0.1
      Backup forwarder: 10.255.0.2

Instance: __default_evpn__
  Route Distinguisher: 10.255.0.1:0
  Encapsulation type: VXLAN
```

```
  VLAN ID: 0
  Per-instance MAC route label: 300208
  Per-instance multicast route label: 300224
  MAC database status                 Local  Remote
    Total MAC addresses:                  0       0
    Default gateway MAC addresses:        0       0
  Number of local interfaces: 0 (0 up)
  Number of IRB interfaces: 0 (0 up)
  Number of neighbors: 1
    Address               MAC    MAC+IP        AD        IM        ES
    10.255.0.2              0         0         0         0         2
  Number of ethernet segments: 0
```

show evpn instance extensive (Preference-based DF Election)

```
user@host> show evpn instance EVPN_1 extensive
```

```
Instance: EVPN_1
  Route Distinguisher: 1:101
  Per-instance MAC route label: 299792
+ DF Election preference: Lowest preference
  MAC database status                     Local  Remote
    MAC advertisements:                       7      15
    MAC+IP advertisements:                    7      12
    Default gateway MAC advertisements:       4       8
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                            Mode             Status
    ae101.0         00:11:11:11:11:11:11:11:11:01  all-active       Up
    ae102.0         00:11:11:11:11:11:11:11:11:02  all-active       Up
  Number of IRB interfaces: 4 (4 up)
    Interface name  VLAN   VNI    Status  L3 context
    irb.101         101             Up     master
    irb.102         102             Up     master
    irb.103         103             Up     master
    irb.104         104             Up     master
  Number of bridge domains: 4
    VLAN  Domain ID   Intfs / up    IRB intf   Mode            MAC sync  IM route
 label
    101                 1    1     irb.101    Extended         Enabled   299776

    102                 1    1     irb.102    Extended         Enabled   299776

    103                 1    1     irb.103    Extended         Enabled   299776

    104                 1    1     irb.104    Extended         Enabled   299776

  Number of neighbors: 2
    100.0.0.2
      Received routes
        MAC address advertisement:             7
        MAC+IP address advertisement:          4
        Inclusive multicast:                   4
        Ethernet auto-discovery:               4
    100.0.0.4
      Received routes
        MAC address advertisement:             8
        MAC+IP address advertisement:          8
        Inclusive multicast:                   4
```

```
      Ethernet auto-discovery:               0
  Number of ethernet segments: 2
    ESI: 00:11:11:11:11:11:11:11:11:11
      Status: Resolved by IFL ae0.110
      Local interface: ae0.110, Status: Up/Forwarding
      Number of remote PEs connected: 2
        Remote PE        MAC label  Aliasing label  Mode
        100.100.100.3    0          0               single-active
        100.100.100.2    0          0               single-active
      DF Election Algorithm: Preference based
      Designated forwarder: 100.100.100.3, Preference: 200
      Backup forwarder: 100.100.100.1, Preference: 800
      Backup forwarder: 100.100.100.2, Preference: 400
    ESI: 00:11:11:11:11:11:11:11:11:02
      Status: Resolved by IFL ae102.0
      Local interface: ae102.0, Status: Up/Forwarding
      Number of remote PEs connected: 1
        Remote PE        MAC label  Aliasing label  Mode
        100.0.0.2        299792     299792          all-active
      DF Election Algorithm: MOD based
      Designated forwarder: 100.0.0.2
      Backup forwarder: 100.0.0.1
      Last designated forwarder update: Feb 21 22:23:32
      Advertised split horizon label: 300800
```

show evpn instance extensive (Duplicate MAC Address)

```
user@host> show evpn instance ALPHA extensive

Instance: ALPHA
  Route Distinguisher: 10.255.0.2:100
  Per-instance MAC route label: 304192
  Duplicate MAC detection threshold: 5
  Duplicate MAC detection window: 180
  Duplicate MAC auto-recovery time: 10

...
```

show evpn instance extensive (Protected Interface)

```
user@host> show evpn instance blue extensive

Instance: blue
  Route Distinguisher: 10.255.255.1:100
  Per-instance MAC route label: 299776
  MAC database status                       Local  Remote
     MAC advertisements:                        0       0
     MAC+IP advertisements:                     0       0
     Default gateway MAC advertisements:        0       0
  Number of local interfaces: 5 (5 up)
     Interface name  ESI                            Mode             Status
AC-Role
     ae0.0           00:11:22:33:44:55:66:77:88:99  all-active       Up
Root
     ge-0/0/3.0      00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
```

```
    ge-0/0/4.0      00:11:11:11:44:55:66:77:88:99  all-active       Up
Root
    ge-0/0/4.1      00:22:22:22:44:55:66:77:88:99  all-active       Up
Root
    ge-0/0/4.50     00:00:00:00:00:00:00:00:00:00  single-homed     Up
Root
  Number of IRB interfaces: 1 (0 up)
    Interface name  VLAN   VNI    Status  L3 context
    irb.1           25             Down   vrf
  Number of protect interfaces: 1
    Interface name  protect-interface   active-interface
    ge-0/0/3.1       ge-0/0/4.50            ge-0/0/3.1
```

show evpn instance esi backup-forwarder (Instance Name with Ethernet Segment Identifier)

```
user@host> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99 backup-forwarder

Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Backup forwarder: 10.255.0.2
```

show evpn instance esi designated-forwarder (Instance Name with Ethernet Segment Identifier)

```
user@host> show evpn instance ALPHA esi 00:11:22:33:44:55:66:77:88:99 designated-forwarder

Instance: ALPHA
  Number of ethernet segments: 1
    ESI: 00:11:22:33:44:55:66:77:88:99
      Designated forwarder: 10.255.0.1
```

show evpn instance (CFM up)

The following shows the partial output listing the interface for show evpn instance command.

```
user@host > show evpn instance evpn_1

Number of local interfaces: 3 (3 up)
    Interface name  ESI                             Mode             Status
AC-Role
    .local..8       00:00:00:00:00:00:00:00:00:00   single-homed     Up
Root
    xe-0/0/11:0.0   00:00:00:00:00:00:00:00:00:00   single-homed     Up
Root
```

show evpn ip-prefix-database

Syntax

```
show evpn ip-prefix-database
<extensive>
<direction (imported | exported)>
<esi number>
<ethernet-tag number>
<family (inet | inet6)>
<gateway ip-address>
<l3-context routing-instance-name>
<logical-system logical-system-name>
<nexthop ip-address>
<prefix ip-prefix>
```

Release Information

Command introduced in Junos OS Release 15.1X53-D30 for QFX10002 switches.
Statement introduced in Junos OS Release 15.1X53-D60 for QFX10008 and QFX10016 switches.
Statement introduced in Junos OS Release 17.3R1 for EX9200 switches.

Description

Display Ethernet VPN (EVPN) database information for imported and exported IP prefixes.

Options

none—Display standard information for all EVPN imported and exported IP prefixes.

extensive—Display the specified level of output.

direction (imported | exported)—(Optional) Display the specified subset of IP prefixes.

esi *number*—(Optional) Display the IP prefix associated with the specified Ethernet segment identifier.

ethernet-tag *number*—(Optional) Display the IP prefix associated with the specified Ethernet tag.

family (inet | inet6)—(Optional) Display IP prefixes for the specified protocol family.

gateway *ip-address*—(Optional) Display the IP Prefix associated with the overlay gateway IP address.

l3-context *routing-instance-name*—(Optional) Display EVPN IP prefix information for the specified Layer 3 virtual routing and forwarding (VRF) instance.

logical-system <*logical-system-name*>—(Optional) Display EVPN IP prefix information for the specified logical system or all logical systems.

nexthop *ip-address*—(Optional) Display EVPN IP prefix information for the specified underlay next-hop IP address.

prefix *ip-prefix*—(Optional) Display EVPN database information for the specified IP prefix.

Required Privilege Level view

Related Documentation
- show evpn l3-context on page 1181
- show route table on page 1217

List of Sample Output
show evpn ip-prefix-database on page 1175
show evpn ip-prefix-database extensive on page 1176

Output Fields Table 36 on page 1174 lists the output fields for the **show evpn ip-prefix-database** command. Output fields are listed in the approximate order in which they appear.

Table 36: show evpn ip-prefix-database Output Fields

Field Name	Field Description	Level of Output
L3 context	Name of virtual routing and forwarding (VRF) instance.	All levels
EVPN Exported Prefixes Prefix	List of exported IP prefixes.	All levels
EVPN route status	For exported prefixes only, status of route: **Created**.	level-of-output none
EVPN imported Prefixes	List of imported EVPN IP prefixes.	All levels
Etag	Ethernet tag	level-of-output none
Route distinguisher	IP address identifier for the IP prefix route.	All levels
VNI	Virtual network identifier for the Layer 3 virtual and routing forwarding (VRF) for the customer or tenant domain.	All levels
Router MAC	MAC address associated with the IP prefix.	All levels
Nexthop/Overlay GW/ESI	For imported IP prefixes, next-hop IP address.	level-of-output none
Change flags	Trace flags.	level-ot-output extensive
Advertisement mode	For exported IP prefixes, type of next-hop address.	level-of-output extensive
Encapsulation	For exported IP prefixes, type of encapsulation	level-of-output extensive
Remote Advertisements	For imported IP prefixes, route distinguisher identifier, MAC address, virtual network identifier, and BGP next-hop address to remote destination.	level-of-output extensive

Sample Output

show evpn ip-prefix-database

```
user@host > show evpn ip-prefix-database

L3 context: VRF-100

IPv4->EVPN Exported Prefixes
Prefix                                  EVPN route status
100.1.0.0/22                            Created
100.1.4.0/22                            Created
100.1.8.0/22                            Created
100.1.12.0/22                           Created
100.1.16.0/22                           Created

IPv6->EVPN Exported Prefixes
Prefix                                  EVPN route status
1234:100:1::/64                         Created
1234:100:1:4::/64                       Created
1234:100:1:8::/64                       Created
1234:100:1:12::/64                      Created
1234:100:1:16::/64                      Created

EVPN->IPv4 Imported Prefixes
Prefix                                  Etag      IP route status
100.2.0.0/22                            0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
100.2.4.0/22                            0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
100.2.8.0/22                            0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
100.2.12.0/22                           0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
100.2.16.0/22                           0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2

EVPN->IPv6 Imported Prefixes
Prefix                                  Etag      IP route status
1234:100:2::/64                         0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
1234:100:2:4::/64                       0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
1234:100:2:8::/64                       0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
```

```
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
1234:100:2:12::/64                           0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
1234:100:2:16::/64                           0         Created
  Route distinguisher    VNI/Label  Router MAC         Nexthop/Overlay GW/ESI
  10.255.2.1:100         9101       00:05:86:e1:03:f0  10.255.2.1
  10.255.2.2:100         9101       00:05:86:d0:a6:f0  10.255.2.2
```

show evpn ip-prefix-database extensive

```
user@host> show evpn ip-prefix-database extensive

L3 context: VRF-100

IPv4->EVPN Exported Prefixes

Prefix: 100.1.0.0/22
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.4.0/22
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.8.0/22
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.12.0/22
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 100.1.16.0/22
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0
```

```
IPv6->EVPN Exported Prefixes

Prefix: 1234:100:1::/64
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:4::/64
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:8::/64
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:12::/64
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

Prefix: 1234:100:1:16::/64
  EVPN route status: Created
  Change flags: 0x0
  Advertisement mode: Direct nexthop
  Encapsulation: VXLAN
  VNI: 9100
  Router MAC: 00:05:86:28:90:f0

EVPN->IPv4 Imported Prefixes

Prefix: 100.2.0.0/22, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 100.2.4.0/22, Ethernet tag: 0
```

```
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 100.2.8.0/22, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 100.2.12.0/22, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 100.2.16.0/22, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

EVPN->IPv6 Imported Prefixes

Prefix: 1234:100:2::/64, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
```

```
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 1234:100:2:4::/64, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 1234:100:2:8::/64, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 1234:100:2:12::/64, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2

Prefix: 1234:100:2:16::/64, Ethernet tag: 0
  IP route status: Created
  Change flags: 0x0
  Remote advertisements:
    Route Distinguisher: 10.255.2.1:100
      VNI: 9101
      Router MAC: 00:05:86:e1:03:f0
      BGP nexthop address: 10.255.2.1
    Route Distinguisher: 10.255.2.2:100
      VNI: 9101
      Router MAC: 00:05:86:d0:a6:f0
      BGP nexthop address: 10.255.2.2
```

show evpn l3-context

Syntax

```
show evpn l3-context
<brief | extensive>
<logical-system logical-system-name>
<routing-instance-name>
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D30 for QFX10002 switches.
Statement introduced in Junos OS Release 15.1X530-D60 for QFX10008 and QFX10016 switches in Junos OS Release 15.1X53-D60.
Statement introduced in Junos OS Release 17.3R1 for EX9200 switches.

Description

Display EVPN database information for Layer 3 virtual routing and forwarding (VRF) instances.

Options

none—Display standard EVPN database information for all Layer 3 VRF instances.

brief | extensive—(Optional) Display the specified level of output.

logical-system <*logical-system-name*>—(Optional) Display Layer 3 information for the specified logical system or all logical systems.

routing-instance-name—(Optional) Display EVPN database information for the specified Layer 3 VRF instance.

Additional Information

Required Privilege Level

view

Related Documentation

- show evpn ip-prefix-database on page 1173

List of Sample Output

show evpn l3-context on page 1182
show evpn l3-context extensive on page 1183

Output Fields

Table 37 on page 1181 lists the output fields for the **show evpn l3-context** command. Output fields are listed in the approximate order in which they appear.

Table 37: show evpn l3-context Output Fields

Field Name	Field Description	Level of Output
L3 context	Name of VRF instance.	All levels
Type	Status of VRF instance	All levels
Fwd	Type of forwarding route	level-of-output none

Table 37: show evpn l3-context Output Fields (continued)

Field Name	Field Description	Level of Output
Encap	Type of encapsulation.	level-of-output none
VNI	Virtual network identifier for VRF instance.	All levels
Router MAC/GW intf	For a pure type-5 route, the router MAC provides next-hop reachability information and the MAC address of the sending device For a standard type-5 route, the GW interface provides the next-hop route.	All levels
Advertisement mode	Type of forwarding route.	level-of-output Extensive
IPv4 source VTEP address	IPv4 source address for the Virtual Extensible LAN (VXLAN) tunnel.	level-of-output Extensive
IPv6 source VTEP adddress	IPv6 source address for the Virtual Extensible LAN (VXLAN) tunnel.	level-of-output Extensive
IP->EVPN export policy	Name of export routing policy applied to forwarded IP routes.	level-of-output extensive
Flags	Enabled trace flags.	level-of-output extensive
Change flags	Changed trace flags.	level-of-output extensive
Composite nexthop	Status of next-hop route: **Enabled** or **Disabled**.	level-of-output extensive
Route distinguisher	Route distinguisher identifier of the VRF instance	level-of-output extensive
Reference count		level-of-output extensive

Sample Output

show evpn l3-context

```
user@DC1_SPINE1_RE> show evpn l3-context

L3 context                     Type  Fwd   Encap  VNI/Label  Router MAC/GW intf
VRF-100                        Cfg   Sym   VXLAN  9100       00:05:86:28:90:f0
VRF-200                        Cfg   Sym   VXLAN  9200       00:05:86:28:90:f0
```

show evpn l3-context extensive

```
user@DC1_SPINE1_RE> show evpn l3-context extensive

L3 context: VRF-100
Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 00:05:86:28:90:f0
  Encapsulation: VXLAN, VNI: 9100
  IPv4 source VTEP address: 10.255.1.1
  IPv6 source VTEP address: abcd::128:102:242:145
  IP->EVPN export policy: EVPN-TYPE5-EXPORT-VRF-100
  Flags: 0x19 <Configured IRB-MAC New>
  Change flags: 0xe <Export-Policy Fwd-Mode Encap>
  Composite nexthop support: Enabled
  Route Distinguisher: 10.255.1.1:100
  Reference count: 21

L3 context: VRF-200
  Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 00:05:86:28:90:f0
  Encapsulation: VXLAN, VNI: 9200
  IPv4 source VTEP address: 10.255.1.1
  IPv6 source VTEP address: abcd::128:102:242:145
  IP->EVPN export policy: EVPN-TYPE5-EXPORT-VRF-200
  Flags: 0x19 <Configured IRB-MAC New>
  Change flags: 0xe <Export-Policy Fwd-Mode Encap>
  Composite nexthop support: Enabled
  Route Distinguisher: 10.255.1.1:200
  Reference count: 21
```

show evpn mac-table

Syntax

```
show evpn mac-table
<age>
<address>
<brief |count | detail | extensive | summary>
<instance instance-name>
<interface interface-name>
<isid <isid>>
<logical-system <logical-system-name>>
<vlan-id vlan-id>
```

Release Information Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Show Ethernet VPN (EVPN) MAC table information.

Options

none—Display brief information about the EVPN MAC table.

age— (Optional) Display age of a single mac-address.

address—(Optional) Display MAC table information for the specified MAC address.

brief | count | detail |extensive | summary—(Optional) Display the specified level of output.

instance *instance-name*—(Optional) Display MAC table information for a specific routing instance.

isid <*isid*>—(Optional) Display MAC table information for the specified ISID or all ISIDs.

logical-system <*logical-system-name*>—(Optional) Display MAC table information for the specified logical system or all logical systems.

vlan-id *vlan-id*—(Optional) Display MAC table information for the specified VLAN.

Required Privilege Level view

Related Documentation

- Example: Configuring an EVPN with IRB Solution on EX9200 Switches on page 636

show evpn nd-table

Syntax

```
show evpn nd-table
<address>
<brief |count | detail | extensive>
<instance instance-name>
```

Release Information

Command introduced in Junos OS Release 16.2.
Command introduced in Junos OS Release 17.1 for MX series Routers.
Command introduced in Junos OS Release 17.3R1 for EX series switches.

Description

Show Ethernet VPN (EVPN) Network Discovery Protocol (NDP) entries associated with learned MAC addresses.

Options

none—Display brief information about the EVPN NDP table.

address—(Optional) Display NDP information for the specified MAC address.

brief | count | detail |extensive—(Optional) Display the specified level of output.

instance <*instance-name*>—(Optional) Display NDP information for the specified routing instance .

Required Privilege Level

view

List of Sample Output

show evpn nd-table on page 1186
show evpn nd-table 00:05:86:a0:dc:f0 (MAC address) on page 1186
show evpn nd-table brief on page 1186
show evpn nd-table detail on page 1186
show evpn nd-table count on page 1187
show evpn and-table extensive on page 1187
show evpn ndp-table instance evpn1 on page 1187

Output Fields

Table 38 on page 1185 lists the output fields for the **show evpn nd-table** command. Output fields are listed in the approximate order in which they appear.

Table 38: show evpn nd-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the NDP table.	All levels
MAC address	MAC addresses learned through NDP.	**brief**, **detail**, **extensive**, **instance**, **mac-address**,,

Table 38: show evpn nd-table Output Fields (continued)

Field Name	Field Description	Level of Output
Logical Interface	Logical interface associated with the routing instance in which the NDP INET address is learned.	**brief**, **instance**, **mac-address**,,
Routing instance	Routing instance in which the NDP INET address is learned.	all levels
Bridging domain	Bridging domain in which the NDP INET address is learned.	all levels
Learning interface	Interface on which the NDP INET address is learned.	**detail**, **extensive**
Count	Indicates the number of NDP INET addresses learned in a routing instance in a bridge domain.	**count**

Sample Output

show evpn nd-table

```
user@host> show evpn nd-table

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8001::2            00:05:86:a0:dc:f0 irb.0         evpn_1       __evpn_1__
```

show evpn nd-table 00:05:86:a0:dc:f0 (MAC address)

```
user@host> show evpn nd-table 00:05:86:a0:dc:f0

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8001::2         00:05:86:a0:dc:f0  irb.0         evpn1        __evpn1__
```

show evpn nd-table brief

```
user@host> show evpn nd-table brief

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8001::2         00:05:86:a0:dc:f0  irb.0         evpn1        __evpn1__
```

show evpn nd-table detail

```
user@host> show evpn nd-table detail


INET address: 8001::2
MAC address: 00:05:86:a0:dc:f0
  Routing instance: evpn1
   Bridging domain: __evpn1__
   Learning interface: irb.0
```

show evpn nd-table count

```
user@switch> show evpn nd-table count

1 NDP INET addresses learned in routing instance evpn1 bridge domain __evpn1__
```

show evpn and-table extensive

```
user@host>  show evpn nd-table extensive


INET address: 8001::2
MAC address: 00:05:86:a0:dc:f0
  Routing instance: evpn1
   Bridging domain: __evpn1__
   Learning interface: irb.0
```

show evpn ndp-table instance evpn1

```
user@host>  show evpn arp-table instance evpn1

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8001::2           00:05:86:a0:dc:f0  irb.0         evpn1        __evpn1__
```

show evpn p2mp

Syntax

```
show evpn p2mp
<brief | extensive>
<instance instance-name>
<logical-system (all | logical-system-name)>
```

Release Information Command introduced in Junos OS Release 18.2.

Description Displays EVPN P2MP information.

Options

brief | detail | extensive—(Optional) Display the specified level of output.

instance *instance-name*—(Optional) Display routing instance information for the specified instance only.

logical-system (all | *logical-system-name*)—(Optional) Display the EVPN P2MP information of all logical systems or a particular logical system.

Required Privilege Level View

List of Sample Output

show evpn p2mp on page 1189
show evpn p2mp extensive on page 1189

Output Fields Table 39 on page 1188 lists the output fields for the show evpn instance command. Output fields are listed in the approximate order in which they appear.

Table 39: show evpn p2mp Output Fields

Field Name	Field Description
Instance	Names of the routing instances.
VLAN	VLAN identifier. The following information is displayed for each VLAN when the **extensive** option is used. • Multicast Nexthop—The Inclusive Multicast (IM) nexthop that is used to send BUM traffic for a VLAN. • P2MP Tunnel—The name of the P2MP tunnel. • PMSI—The provider multicast service interface attributes as advertised in EVPN IM route. This is described in RFC 7432. • ESI—Ethernet segment identifier (ESI) value of the interfaces. For each ESI, the SH nexhop identifier that is used.

Table 39: show evpn p2mp Output Fields (continued)

Field Name	Field Description
Neighbor Address	IP address of the EVPN neighbor. The following information is displayed for each neighbor when the **extensive** option is used. • Transport Labels—The LDP/RSVP transport label for the P2MP LSP rooted at this neighbor • PMSI—The provider multicast service interface attributes as advertised in EVPN IM route. This is described in RFC 7432.
Multicast Nexthop	Multicast nexthop identifier. This is the Inclusive Multicast (IM) nexthop that is used to send BUM traffic for a VLAN
E-tree Leaf Nexthop	E-tree leaf nexthop identifier.
Remote PE	IP address of the remote provider edge router.
Transport Labels	The LDP/RSVP transport label for the P2MP LSP for the remote PE.

Sample Output

show evpn p2mp

```
user@host> show evpn p2mp
```

```
regress@PE1_re> show evpn p2mp
Instance: ALPHA
  VLAN       Multicast Nexthop   E-tree Leaf Nexthop
  100        1048576
    Remote PE         VLAN     Transport Labels
    10.255.0.2        100      300448
    10.255.0.3        100      300560
    10.255.0.4        100      300608
```

show evpn p2mp extensive

```
user@host> show evpn p2mp extensive
```

```
Instance: ALPHA
  VLAN ID: 100
    Multicast Nexthop: 1048576
      P2MP Tunnel: 10.255.0.1:100:ldp-p2mp:evpn:100:ALPHA
      PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.1 , lsp-id 16777217
        ESI: 00:11:22:33:44:55:66:77:88:99 SH Nexthop: 1048577
  Neighbor Address: 10.255.0.2, VLAN ID: 100
    Transport Labels: 300448
      PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.2 , lsp-id 16777217
  Neighbor Address: 10.255.0.3, VLAN ID: 100
    Transport Labels: 300560
      PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.3 , lsp-id 16777217
  Neighbor Address: 10.255.0.4, VLAN ID: 100
```

```
Transport Labels: 300608
  PMSI: Flags 0x0: Label 0: LDP-P2MP: Root 10.255.0.4 , lsp-id 16777217
```

show evpn peer-gateway-macs

Syntax

```
show evpn peer-gateway-macs
<address>
<instance instance-name>
```

Release Information Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Show Ethernet VPN (EVPN) peer gateway MAC information.

Options

none—Display brief information about the EVPN peer gateway MAC.

address—(Optional) Display peer gateway information for the specified MAC address.

instance *instance-name*—(Optional) Display peer gateway MAC information for the specified routing instance.

Required Privilege Level view

Related Documentation

- Example: Configuring an EVPN with IRB Solution on EX9200 Switches on page 636

show evpn prefix

Syntax show evpn prefix

Release Information Command introduced in Junos OS Release 17.1 on MX Series routers with MPCs.

Description Display Ethernet VPN (EVPN) information for imported and exported prefixes.

Required Privilege Level view

List of Sample Output show evpn prefixes on page 1192

Output Fields The table 2 lists the output fields for the **show evpn ip-prefix-database** command. Output fields are listed in the approximate order in which they appear.

Table 40: show evpn prefix Output Fields

Field Name	Field Description
VLAN	VLAN identifier of the Layer 2 circuit.
VNI	VXLAN network identifier for the Layer 3 virtual and routing forwarding (VRF) for the customer or tenant domain.
Prefix	MAC address associated with the IP prefix.
Active Source	The IP address or the ESI value or the IRB interface name of the source.
Timestamp	Month, day and time of generated command output.

Sample Output

show evpn prefixes

```
user@host > show evpn prefixes

Instance: ALPHA
VLAN  VNI    Prefix                    Active source                  Timestamp

      9100   10.0.0.0/24               10.255.0.2                     May 22
17:16:12
      9100   20.0.0.0/24               00:11:22:33:44:55:66:77:88:99  May 22
22:33:35
      9100   30.0.0.0/24               irb.0                          May 22
17:13:31
      9100   40.0.0.0/24               10.255.0.3                     May 22
17:14:20
```

show evpn vpws-instance

Syntax

show evpn vpws-instance
evpn-vpws-instance-name

Release Information

Statement introduced in Junos OS Release 17.1 for MX Series routers.

Description

Display the details of the VPWS instance of the EVPN.

Options

evpn-vpws-instance-name—(Optional) Name of the VPWS instance of the EVPN.

Required Privilege Level

view

Related Documentation

- Overview of VPWS with EVPN Signaling Mechanisms on page 666

Output Fields

Table 41 on page 1193 describes the output fields for the **show evpn vpws-instance** command. Output fields are listed in the approximate order in which they appear.

Table 41: show evpn vpws-instance Output Fields

Field Name	Field Description
Instance	Name of the routing instance.
Route Distinguisher	Route distinguisher as configured for the routing instance on the node.
Number of local interfaces	Number of interfaces configured to be part of the routing instance.
Interface name	Name of the interface.
ESI	Ethernet segment identifier (ESI) configured for the interface.
Mode	Single-homed, single-active, or all-active form of multihoming.
Role	Primary or backup, depending on the role. The role, depending on the mode, is as follows: • Single-active multihoming–Role is primary if the interface is the designated forwarder(DF) and backup if the interface is the non-DF. • All-active multihoming–Role is always primary. • Single-homed–Role is always primary.
Status	Status of the interface is either Up or Down. Interface is available if the status is Up and interface is not available if the interface is Down.
Local SID	Local service identifier configured on the interface.

Table 41: show evpn vpws-instance Output Fields (continued)

Field Name	Field Description
Advertised Label	Label advertised for the service identifier as part of the autodiscovery route per the EVPN instance (EVI).
PE routers hosting the local service identifier as configured on the interface	
PE addr	IP address of the PE router advertising the autodiscovered route per the EVI.
ESI	ESI encoded in the autodiscovered route per the EVI.
Label	Label as encoded in the autodiscovered route per the EVI.
Mode	Single-homed, single-active, or all-active form of multihoming.
Role	Primary or backup depending on the role. The role depending on the mode is as follows: • Single-active multihoming–Role is primary if the interface is designated forwarder(DF) and backup if the interface is non-DF. • All-active multihoming–Role is always primary. • Single-homed–Role is always primary.
TS	Timestamp of the receipt of the autodiscovered route per the EVI. This is used as tie breaker in case there are two PE routers advertising the autodiscovered route per the EVI with the P bit set.
Status	The status can be as follows: • Resolved—The autodiscovered route per the ESI was received for the ESI from the PE router. • Unresolved—The autodiscovered route per the ESI was not received for the ESI from the PE router.
PE routers hosting the remote service identifier as configured on the interface	
Remote SID	Remote service identifier configured on the interface.
PE addr	IP address of the PE router advertising the autodiscovered route per the EVI.
ESI	ESI encoded in the autodiscovered route per the EVI.
Label	Label as encoded in the autodiscovery route per the EVI.
Mode	Single-homed, single-active, or all-active form of multihoming.

Table 41: show evpn vpws-instance Output Fields (continued)

Field Name	Field Description
Role	Primary or backup depending on the role. The role depending on the mode is as follows: • Single-active multihoming–Role is primary if the interface is designated forwarder(DF) and backup if the interface is non-DF. • All-active multihoming–Role is always primary. • Single-homed–Role is always primary.
TS	Timestamp of the receipt of the autodiscovered route per the EVI. This is used as tie breaker in case there are two PE routers advertising the autodiscovered route per the EVI with the P bit set.
Status	The status can be as follows: • Resolved—The autodiscovered route per the ESI was received for the ESI from the PE router. • Unresolved—The autodiscovered route per the ESI was not received for the ESI from the PE router.
Fast Convergence Information	Details of the ESI, the PE router, and the advertised service identifier used during fast convergence.
ESI	ESI as advertised in the autodiscovered route per the ESI.
Number of PE nodes	Number of PE nodes available.
PE	IP address of the PE router advertising the autodiscovered route per the ESI.
Advertised SID	List of service identifiers that get impacted if the PE router withdraws the autodiscovered route per the ESI.
DF Election Information for Single-Active ESI	Details of the DF election information for the single-active ESI.
ESI	ESI as advertised in the Ethernet segment route.
SID used for DF Election	Minimum service identifier configured for the ESI in the EVI.
Primary PE	PE router that is the DF for the ESI.
Backup PE	PE router that is the backup DF for the ESI.
Last DF Election	Last time when the DF election algorithm was executed on the PE router.

Sample Output

```
user@host> show evpn vpws-instance vpws1004
```

```
Instance: vpws1004
  Route Distinguisher: 100.100.100.4:1004
  Number of local interfaces: 1 (1 up)

    Interface name  ESI                             Mode            Role
Status
    ge-0/0/1.1004   00:00:00:00:00:00:00:00:00:00  single-homed    Primary    Up

        Local SID: 1004 Advertised Label: 301360
        Remote SID: 2004
            PE addr         ESI                             Label  Mode
Role     TS                      Status
            100.100.100.2   00:11:11:11:11:11:11:11:11:11 301888 all-active
Primary  2017-01-11 21:57:31.916 Resolved
            100.100.100.1   00:11:11:11:11:11:11:11:11:11 301952 all-active
Primary  2017-01-11 21:59:28.491 Resolved
            100.100.100.3   00:11:11:11:11:11:11:11:11:11 301984 all-active
Primary  2017-01-11 21:59:28.522 Resolved

    Fast Convergence Information
    ESI: 00:11:11:11:11:11:11:11:11:11 Number of PE nodes: 3
        PE: 100.100.100.2
            Advertised SID: 2004
        PE: 100.100.100.1
            Advertised SID: 2004
        PE: 100.100.100.3
            Advertised SID: 2004
```

user@host>**show evpn vpws-instance vpws1003**

```
Instance: vpws1003
  Route Distinguisher: 100.100.100.4:1003
  Number of local interfaces: 2 (2 up)

    Interface name  ESI                             Mode            Role
Status
    ae10.2003       00:44:44:44:44:44:44:44:44:44  all-active      Primary    Up

        Local SID: 2003 Advertised Label: 301312
            PE addr         ESI                             Label  Mode
Role     TS                      Status
            100.100.100.3   00:44:44:44:44:44:44:44:44:44 301792 all-active
Primary  2017-01-11 21:59:28.498 Resolved
        Remote SID: 1003
            Local Interface: ge-0/0/1.1003 (Up)

    Interface name  ESI                             Mode            Role
Status
    ge-0/0/1.1003   00:00:00:00:00:00:00:00:00:00  single-homed    Primary    Up

        Local SID: 1003 Advertised Label: 301296
        Remote SID: 2003
            Local Interface: ae10.2003 (Up)
            PE addr         ESI                             Label  Mode
Role     TS                      Status
            100.100.100.3   00:44:44:44:44:44:44:44:44:44 301792 all-active
Primary  2017-01-11 21:59:28.498 Resolved
```

```
Fast Convergence Information
ESI: 00:44:44:44:44:44:44:44:44:44 Number of PE nodes: 1
    PE: 100.100.100.3
        Advertised SID: 2003
```

show route forwarding-table

List of Syntax

Syntax

```
show route forwarding-table
<detail | extensive | summary>
<all>
<ccc interface-name>
<destination destination-prefix>
<family family | matching matching>
<interface-name interface-name>
<label name>
<matching matching>
<multicast>
<table (default | logical-system-name/routing-instance-name | routing-instance-name)>
<vlan (all | vlan-name)>
<vpn vpn>
```

Syntax (MX Series Routers)

```
show route forwarding-table
<detail | extensive | summary>
<all>
<bridge-domain (all | domain-name)>
<ccc interface-name>
<destination destination-prefix>
<family family | matching matching>
<interface-name interface-name>
<label name>
<learning-vlan-id learning-vlan-id>
<matching matching>
<multicast>
<table (default | logical-system-name/routing-instance-name | routing-instance-name)>
<vlan (all | vlan-name)>
<vpn vpn>
```

Syntax (TX Matrix and TX Matrix Plus Routers)

```
show route forwarding-table
<detail | extensive | summary>
<all>
<ccc interface-name>
<destination destination-prefix>
<family family | matching matching>
<interface-name interface-name>
<matching matching>
<label name>
<lcc number>
<multicast>
<table routing-instance-name>
<vpn vpn>
```

Release Information Command introduced before Junos OS Release 7.4.
Option **bridge-domain** introduced in Junos OS Release 7.5
Option **learning-vlan-id** introduced in Junos OS Release 8.4
Options **all** and **vlan** introduced in Junos OS Release 9.6.
Command introduced in Junos OS Release 11.3 for the QFX Series.
Command introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Display the Routing Engine's forwarding table, including the network-layer prefixes and their next hops. This command is used to help verify that the routing protocol process has relayed the correction information to the forwarding table. The Routing Engine constructs and maintains one or more routing tables. From the routing tables, the Routing Engine derives a table of active routes, called the forwarding table.

NOTE: The Routing Engine copies the forwarding table to the Packet Forwarding Engine, the part of the router that is responsible for forwarding packets. To display the entries in the Packet Forwarding Engine's forwarding table, use the **show pfe route** command.

Options

none—Display the routes in the forwarding tables. By default, the **show route forwarding-table** command does not display information about private, or internal, forwarding tables.

detail | extensive | summary—(Optional) Display the specified level of output.

all—(Optional) Display routing table entries for all forwarding tables, including private, or internal, tables.

bridge-domain (all | *bridge-domain-name*)—(MX Series routers only) (Optional) Display route entries for all bridge domains or the specified bridge domain.

ccc *interface-name*—(Optional) Display route entries for the specified circuit cross-connect interface.

destination *destination-prefix*—(Optional) Destination prefix.

family *family*—(Optional) Display routing table entries for the specified family: **bridge (ccc | destination | detail | extensive | interface-name | label | learning-vlan-id | matching | multicast | summary | table | vlan | vpn)**, **ethernet-switching**, **evpn**, **fibre-channel**, **fmembers**, **inet**, **inet6**, **iso**, **mcsnoop-inet**, **mcsnoop-inet6**, **mpls**, **satellite-inet**, **satellite-inet6**, **satellite-vpls**, **tnp**, **unix**, **vpls**, or **vlan-classification**.

interface-name *interface-name*—(Optional) Display routing table entries for the specified interface.

label *name*—(Optional) Display route entries for the specified label.

lcc *number*—(TX Matrix and TX matrix Plus routers only) (Optional) On a routing matrix composed of a TX Matrix router and T640 routers, display information for the

specified T640 router (or line-card chassis) connected to the TX Matrix router. On a routing matrix composed of the TX Matrix Plus router and T1600 or T4000 routers, display information for the specified router (line-card chassis) connected to the TX Matrix Plus router.

Replace *number* with the following values depending on the LCC configuration:

- 0 through 3, when T640 routers are connected to a TX Matrix router in a routing matrix.
- 0 through 3, when T1600 routers are connected to a TX Matrix Plus router in a routing matrix.
- 0 through 7, when T1600 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix.
- 0, 2, 4, or 6, when T4000 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix.

learning-vlan-id *learning-vlan-id*—(MX Series routers only) (Optional) Display learned information for all VLANs or for the specified VLAN.

matching *matching*—(Optional) Display routing table entries matching the specified prefix or prefix length.

multicast—(Optional) Display routing table entries for multicast routes.

table —(Optional) Display route entries for all the routing tables in the main routing instance or for the specified routing instance. If your device supports logical systems, you can also display route entries for the specified logical system and routing instance. To view the routing instances on your device, use the **show route instance** command.

vlan (all | *vlan-name*)—(Optional) Display information for all VLANs or for the specified VLAN.

vpn *vpn*—(Optional) Display routing table entries for a specified VPN.

Required Privilege Level view

List of Sample Output

show route forwarding-table on page 1205
show route forwarding-table detail on page 1206
show route forwarding-table destination extensive (Weights and Balances) on page 1207
show route forwarding-table extensive on page 1207
show route forwarding-table extensive (RPF) on page 1208
show route forwarding-table (dynamic list next hop) on page 1209
show route forwarding-table family mpls on page 1210
show route forwarding-table family mpls ccc ge-0/0/1.1004 on page 1210
show route forwarding-table family vpls on page 1211
show route forwarding-table vpls (Broadcast, unknown unicast, and multicast (BUM) hashing is enabled) on page 1211

show route forwarding-table vpls (Broadcast, unknown unicast, and multicast (BUM) hashing is enabled with MAC Statistics) on page 1211
show route forwarding-table family vpls extensive on page 1212
show route forwarding-table table default on page 1213
show route forwarding-table table logical-system-name/routing-instance-name on page 1214
show route forwarding-table vpn on page 1215

Output Fields Table 42 on page 1201 lists the output fields for the **show route forwarding-table** command. Output fields are listed in the approximate order in which they appear. Field names might be abbreviated (as shown in parentheses) when no level of output is specified, or when the **detail** keyword is used instead of the **extensive** keyword.

Table 42: show route forwarding-table Output Fields

Field Name	Field Description	Level of Output
Logical system	Name of the logical system. This field is displayed if you specify the **table** ***logical-system-name/routing-instance-name*** option on a device that is configured for and supports logical systems.	All levels
Routing table	Name of the routing table (for example, inet, inet6, mpls).	All levels

Table 42: show route forwarding-table Output Fields (continued)

Field Name	Field Description	Level of Output
Enabled protocols	The features and protocols that have been enabled for a given routing table. This field can contain the following values: • BUM hashing—BUM hashing is enabled. • MAC Stats—Mac Statistics is enabled. • Bridging—Routing instance is a normal layer 2 bridge. • No VLAN—No VLANs are associated with the bridge domain. • All VLANs—The **vlan-id all** statement has been enabled for this bridge domain. • Single VLAN—Single VLAN ID is associated with the bridge domain. • MAC action drop—New MACs will be dropped when the MAC address limit is reached. • Dual VLAN—Dual VLAN tags are associated with the bridge domain • No local switching—No local switching is enabled for this routing instance.. • Learning disabled—Layer 2 learning is disabled for this routing instance. • MAC limit reached—The maximum number of MAC addresses that was configured for this routing instance has been reached. • VPLS—The VPLS protocol is enabled. • No IRB l2-copy—The no-irb-layer-2-copy feature is enabled for this routing instance. • ACKed by all peers—All peers have acknowledged this routing instance. • BUM Pruning—BUM pruning is enabled on the VPLS instance. • Def BD VXLAN—VXLAN is enabled for the default bridge domain. • EVPN—EVPN protocol is enabled for this routing instance. • Def BD OVSDB—Open vSwitch Database (OVSDB) is enabled on the default bridge domain. • Def BD Ingress replication—VXLAN ingress node replication is enabled on the default bridge domain. • L2 backhaul—Layer 2 backhaul is enabled. • FRR optimize—Fast reroute optimization • MAC pinning—MAC pinning is enabled for this bridge domain. • MAC Aging Timer—The MAC table aging time is set per routing instance. • EVPN VXLAN—This routing instance supports EVPN with VXLAN encapsulation. • PBBN—This routing instance is configured as a provider backbone bridged network. • PBN—This routing instance is configured as a provider bridge network. • ETREE—The ETREE protocol is enabled on this EVPN routing instance. • ARP/NDP suppression—EVPN ARP NDP suppression is enabled in this routing instance. • Def BD EVPN VXLAN—EVPN VXLAN is enabled for the default bridge domain. • MPLS control word—Control word is enabled for this MPLS routing instance.	All levels
Address family	Address family (for example, **IP**, **IPv6**, **ISO**, **MPLS**, and **VPLS**).	All levels
Destination	Destination of the route.	**detail extensive**

Table 42: show route forwarding-table Output Fields (continued)

Field Name	Field Description	Level of Output
Route Type (Type)	How the route was placed into the forwarding table. When the **detail** keyword is used, the route type might be abbreviated (as shown in parentheses): • **cloned** (**clon**)—(TCP or multicast only) Cloned route. • **destination** (**dest**)—Remote addresses directly reachable through an interface. • **destination down** (**iddn**)—Destination route for which the interface is unreachable. • **interface cloned** (**ifcl**)—Cloned route for which the interface is unreachable. • **route down** (**ifdn**)—Interface route for which the interface is unreachable. • **ignore** (**ignr**)—Ignore this route. • **interface** (**intf**)—Installed as a result of configuring an interface. • **permanent** (**perm**)—Routes installed by the kernel when the routing table is initialized. • **user**—Routes installed by the routing protocol process or as a result of the configuration.	All levels
Route Reference (RtRef)	Number of routes to reference.	**detail extensive**
Flags	Route type flags: • **none**—No flags are enabled. • **accounting**—Route has accounting enabled. • **cached**—Cache route. • **incoming-iface** ***interface-number***—Check against incoming interface. • **prefix load balance**—Load balancing is enabled for this prefix. • **rt nh decoupled**—Route has been decoupled from the next hop to the destination. • **sent to PFE**—Route has been sent to the Packet Forwarding Engine. • **static**—Static route.	**extensive**
Next hop	IP address of the next hop to the destination.	**detail extensive**

Table 42: show route forwarding-table Output Fields (continued)

Field Name	Field Description	Level of Output
Next hop Type (Type)	Next-hop type. When the **detail** keyword is used, the next-hop type might be abbreviated (as indicated in parentheses): • **broadcast (bcst)**—Broadcast. • **deny**—Deny. • **discard (dscd)** —Discard. • **hold**—Next hop is waiting to be resolved into a unicast or multicast type. • **indexed (idxd)**—Indexed next hop. • **indirect (indr)**—Indirect next hop. • **local (locl)**—Local address on an interface. • **routed multicast (mcrt)**—Regular multicast next hop. • **multicast (mcst)**—Wire multicast next hop (limited to the LAN). • **multicast discard (mdsc)**—Multicast discard. • **multicast group (mgrp)**—Multicast group member. • **receive (recv)**—Receive. • **reject (rjct)**—Discard. An ICMP unreachable message was sent. • **resolve (rslv)**—Resolving the next hop. • **unicast (ucst)**—Unicast. • **unilist (ulst)**—List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.	**detail extensive**
Index	Software index of the next hop that is used to route the traffic for a given prefix.	**detail extensive** none
Route interface-index	Logical interface index from which the route is learned. For example, for interface routes, this is the logical interface index of the route itself. For static routes, this field is zero. For routes learned through routing protocols, this is the logical interface index from which the route is learned.	**extensive**
Reference (NhRef)	Number of routes that refer to this next hop.	**detail extensive** none
Next-hop interface (Netif)	Interface used to reach the next hop.	**detail extensive** none
Weight	Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible (see the **Balance** field description).	**extensive**
Balance	Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a router is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.	**extensive**
RPF interface	List of interfaces from which the prefix can be accepted. Reverse path forwarding (RPF) information is displayed only when **rpf-check** is configured on the interface.	**extensive**

Sample Output

show route forwarding-table

```
user@host> show route forwarding-table

Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    46     4
0.0.0.0/32         perm     0                    dscd    44     1
172.16.1.0/24         ifdn     0                    rslv   608     1 ge-2/0/1.0
172.16.1.0/32         iddn     0 172.16.1.0           recv   606     1 ge-2/0/1.0
172.16.1.1/32         user     0                    rjct    46     4
172.16.1.1/32         intf     0 172.16.1.1           locl   607     2
172.16.1.1/32         iddn     0 172.16.1.1           locl   607     2
172.16.1.255/32       iddn     0 ff:ff:ff:ff:ff:ff  bcst   605     1 ge-2/0/1.0
10.0.0.0/24        intf     0                    rslv   616     1 ge-2/0/0.0
10.0.0.0/32        dest     0 10.0.0.0           recv   614     1 ge-2/0/0.0
10.0.0.1/32        intf     0 10.0.0.1           locl   615     2
10.0.0.1/32        dest     0 10.0.0.1           locl   615     2
10.0.0.255/32      dest     0 10.0.0.255         bcst   613     1 ge-2/0/0.0
10.1.1.0/24        ifdn     0                    rslv   612     1 ge-2/0/1.0
10.1.1.0/32        iddn     0 10.1.1.0           recv   610     1 ge-2/0/1.0
10.1.1.1/32        user     0                    rjct    46     4
10.1.1.1/32        intf     0 10.1.1.1           locl   611     2
10.1.1.1/32        iddn     0 10.1.1.1           locl   611     2
10.1.1.255/32      iddn     0 ff:ff:ff:ff:ff:ff  bcst   609     1 ge-2/0/1.0
10.206.0.0/16      user     0 10.209.63.254      ucst   419    20 fxp0.0
10.209.0.0/16      user     1 0:12:1e:ca:98:0    ucst   419    20 fxp0.0
10.209.0.0/18      intf     0                    rslv   418     1 fxp0.0
10.209.0.0/32      dest     0 10.209.0.0         recv   416     1 fxp0.0
10.209.2.131/32    intf     0 10.209.2.131       locl   417     2
10.209.2.131/32    dest     0 10.209.2.131       locl   417     2
10.209.17.55/32    dest     0 0:30:48:5b:78:d2   ucst   435     1 fxp0.0
10.209.63.42/32    dest     0 0:23:7d:58:92:ca   ucst   434     1 fxp0.0
10.209.63.254/32   dest     0 0:12:1e:ca:98:0    ucst   419    20 fxp0.0
10.209.63.255/32   dest     0 10.209.63.255      bcst   415     1 fxp0.0
10.227.0.0/16      user     0 10.209.63.254      ucst   419    20 fxp0.0

...

Routing table: iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                   rjct    27     1
47.0005.80ff.f800.0000.0108.0003.0102.5524.5220.00
intf     0                   locl    28     1

Routing table: inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                   rjct     6     1
ff00::/8           perm     0                   mdsc     4     1
ff02::1/128        perm     0 ff02::1           mcst     3     1

Routing table: ccc
MPLS:
Interface.Label    Type RtRef Next hop           Type Index NhRef Netif
```

```
default            perm     0                   rjct 16     1
100004(top)fe-0/0/1.0
```

show route forwarding-table detail

```
user@host> show route forwarding-table detail

Routing table: inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            user     2 0:90:69:8e:b1:1b   ucst   132     4 fxp0.0
default            perm     0                    rjct    14     1
10.1.1.0/24        intf     0 ff.3.0.21          ucst   322     1 so-5/3/0.0
10.1.1.0/32        dest     0 10.1.1.0           recv   324     1 so-5/3/0.0
10.1.1.1/32        intf     0 10.1.1.1           locl   321     1
10.1.1.255/32      dest     0 10.1.1.255         bcst   323     1 so-5/3/0.0
10.21.21.0/24      intf     0 ff.3.0.21          ucst   326     1 so-5/3/0.0
10.21.21.0/32      dest     0 10.21.21.0         recv   328     1 so-5/3/0.0
10.21.21.1/32      intf     0 10.21.21.1         locl   325     1
10.21.21.255/32    dest     0 10.21.21.255       bcst   327     1 so-5/3/0.0
127.0.0.1/32       intf     0 127.0.0.1          locl   320     1
172.17.28.19/32    clon     1 192.168.4.254      ucst   132     4 fxp0.0
172.17.28.44/32    clon     1 192.168.4.254      ucst   132     4 fxp0.0

...

Routing table: private1__.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    46     1
10.0.0.0/8         intf     0                    rslv   136     1 fxp1.0
10.0.0.0/32        dest     0 10.0.0.0           recv   134     1 fxp1.0
10.0.0.4/32        intf     0 10.0.0.4           locl   135     2
10.0.0.4/32        dest     0 10.0.0.4           locl   135     2

...

Routing table: iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    38     1

Routing table: inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    22     1
ff00::/8           perm     0                    mdsc    21     1
ff02::1/128        perm     0 ff02::1            mcst    17     1

...

Routing table: mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct 28     1
```

show route forwarding-table destination extensive (Weights and Balances)

```
user@host> show route forwarding-table destination 3.4.2.1 extensive

Routing table: inet [Index 0]
Internet:

Destination:  3.4.2.1/32
  Route type: user
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: unilist                Index: 262143   Reference: 1
  Nexthop: 172.16.4.4
  Next-hop type: unicast                Index: 335      Reference: 2
  Next-hop interface: so-1/1/0.0        Weight: 22      Balance: 3
  Nexthop: 145.12.1.2
  Next-hop type: unicast                Index: 337      Reference: 2
  Next-hop interface: so-0/1/2.0        Weight: 33      Balance: 33
```

show route forwarding-table extensive

```
user@host> show route forwarding-table extensive

Routing table: inet [Index 0]
Internet:

Destination:  default
  Route type: user
  Route reference: 2                    Route interface-index: 0
  Flags: sent to PFE
  Nexthop: 00:00:5E:00:53:1b
  Next-hop type: unicast                Index: 132      Reference: 4
  Next-hop interface: fxp0.0

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: none
  Next-hop type: reject                 Index: 14       Reference: 1

Destination:  127.0.0.1/32
  Route type: interface
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Nexthop: 127.0.0.1
  Next-hop type: local                  Index: 320      Reference: 1

...

Routing table: private1__.inet [Index 1]
Internet:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 46       Reference: 1
```

```
Destination:  10.0.0.0/8
  Route type: interface
  Route reference: 0                    Route interface-index: 3
  Flags: sent to PFE
  Next-hop type: resolve                Index: 136      Reference: 1
  Next-hop interface: fxp1.0

...

Routing table: iso [Index 0]
ISO:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 38       Reference: 1

Routing table: inet6 [Index 0]
Internet6:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 22       Reference: 1

Destination:  ff00::/8
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: multicast discard      Index: 21       Reference: 1

...

Routing table: private1__.inet6 [Index 1]
Internet6:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 54       Reference: 1

Destination:  fe80::2a0:a5ff:fe3d:375/128
  Route type: interface
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Nexthop: fe80::2a0:a5ff:fe3d:375
  Next-hop type: local                  Index: 75       Reference: 1

...
```

show route forwarding-table extensive (RPF)

The next example is based on the following configuration, which enables an RPF check on all routes that are learned from this interface, including the interface route:

```
so-1/1/0 {
    unit 0 {
        family inet {
            rpf-check;
            address 192.0.2.2/30;
        }
    }
}
```

user@host> **show route forwarding-table extensive**

```
Routing table: inet [Index 0]
Internet:
...
...
Destination:  192.0.2.3/32
  Route type: destination
  Route reference: 0                    Route interface-index: 67
  Flags: sent to PFE
  Nexthop: 192.0.2.3
  Next-hop type: broadcast              Index: 328      Reference: 1
  Next-hop interface: so-1/1/0.0
  RPF interface: so-1/1/0.0
```

show route forwarding-table (dynamic list next hop)

The **show route forwarding table** output shows the two next hop elements for a multihomed EVPN destination.

```
user@host> show route forwarding-table label 299952 extensive
```

```
MPLS:

Destination:  299952
  Route type: user
  Route reference: 0                    Route interface-index: 0
  Multicast RPF nh index: 0
  P2mpidx: 0
  Flags: sent to PFE, rt nh decoupled
  Next-hop type: indirect               Index: 1048575  Reference: 2
  Nexthop:
  Next-hop type: composite              Index: 601      Reference: 2
  Next-hop type: indirect               Index: 1048574  Reference: 3
  Nexthop: 1.0.0.4
  Next-hop type: Push 301632, Push 299776(top) Index: 600 Reference: 2
  Load Balance Label: None
  Next-hop interface: ge-0/0/1.0
  Next-hop type: indirect               Index: 1048577  Reference: 3
  Nexthop: 1.0.0.4
  Next-hop type: Push 301344, Push 299792(top) Index: 619 Reference: 2
  Load Balance Label: None
  Next-hop interface: ge-0/0/1.0
```

After one of the PE router has been disabled in the EVPN multihomed network, the same **show route forwarding table** output command shows one next hop element and one empty next hop element.

```
user@host> show route forwarding-table label 299952 extensive

Routing table: default.mpls [Index 0]
MPLS:

Destination:  299952
  Route type: user
  Route reference: 0                    Route interface-index: 0
  Multicast RPF nh index: 0
  P2mpidx: 0
  Flags: sent to PFE, rt nh decoupled
  Next-hop type: indirect               Index: 1048575  Reference: 2
  Nexthop:
  Next-hop type: composite              Index: 601      Reference: 2
  Next-hop type: indirect               Index: 1048577  Reference: 3
  Nexthop: 1.0.0.4
  Next-hop type: Push 301344, Push 299792(top) Index: 619 Reference: 2
  Load Balance Label: None
  Next-hop interface: ge-0/0/1.0
```

show route forwarding-table family mpls

```
user@host> show route forwarding-table family mpls

Routing table: mpls
MPLS:
Destination        Type RtRef Next hop          Type Index NhRef Netif
default            perm     0                   rjct    19     1
0                  user     0                   recv    18     3
1                  user     0                   recv    18     3
2                  user     0                   recv    18     3
100000             user     0 10.31.1.6         swap  100001     fe-1/1/0.0
800002             user     0                   Pop              vt-0/3/0.32770

vt-0/3/0.32770 (VPLS)
                   user     0                   indr   351     4
                                                Push 800000, Push 100002(top)
so-0/0/0.0
```

show route forwarding-table family mpls ccc ge-0/0/1.1004

```
user@host>show route forwarding-table mpls ccc ge-0/0/1.1004

Routing table: default.mpls
MPLS:
Destination        Type RtRef Next hop          Type  Index    NhRef Netif
ge-0/0/1.1004  (CCC) user     0                    ulst   1048577     2
                                                    comp       754         3
                                                   comp       755         3
                                                comp       756         3

Routing table: __mpls-oam__.mpls
MPLS:
```

```
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      556     1
```

show route forwarding-table family vpls

```
user@host> show route forwarding-table family vpls

Routing table: green.vpls
VPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            dynm     0                    flood   353     1
default            perm     0                    rjct    298     1
fe-0/1/0.0         dynm     0                    flood   355     1
00:00:5E:00:53:1f/48              <<<<<Remote CE

                   dynm     0                    indr   351     4
                                                 Push 800000, Push 100002(top)
so-0/0/0.0
00:00:5E:00:53:1f/48              <<<<<<Local CE

                   dynm     0                    ucst   354     2 fe-0/1/0.0
```

show route forwarding-table vpls (Broadcast, unknown unicast, and multicast (BUM) hashing is enabled)

```
user@host> show route forwarding-table vpls

Routing table: green.vpls
VPLS:
Enabled protocols: BUM hashing
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      519     1
lsi.1048832        intf     0                    indr  1048574     4
                              172.16.3.2          Push 262145      621     2
ge-3/0/0.0
00:00:5E:00:53:01/48 user     0                  ucst      590     5 ge-2/3/9.0
0x30003/51         user     0                    comp      627     2
ge-2/3/9.0         intf     0                    ucst      590     5 ge-2/3/9.0
ge-3/1/3.0         intf     0                    ucst      619     4 ge-3/1/3.0
0x30002/51         user     0                    comp      600     2
0x30001/51         user     0                    comp      597     2
```

show route forwarding-table vpls (Broadcast, unknown unicast, and multicast (BUM) hashing is enabled with MAC Statistics)

```
user@host> show route forwarding-table vpls

Routing table: green.vpls
VPLS:
Enabled protocols: BUM hashing, MAC Stats
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      519     1
lsi.1048834        intf     0                    indr  1048574     4
                              172.16.3.2          Push 262145      592     2
ge-3/0/0.0
```

```
00:19:e2:25:d0:01/48 user     0                 ucst      590     5 ge-2/3/9.0
0x30003/51           user     0                 comp      630     2
ge-2/3/9.0           intf     0                 ucst      590     5 ge-2/3/9.0
ge-3/1/3.0           intf     0                 ucst      591     4 ge-3/1/3.0
0x30002/51           user     0                 comp      627     2
0x30001/51           user     0                 comp      624     2
```

show route forwarding-table family vpls extensive

```
user@host> show route forwarding-table family vpls extensive

Routing table: green.vpls [Index 2]
VPLS:

Destination:  default
  Route type: dynamic
  Route reference: 0                    Route interface-index: 72
  Flags: sent to PFE
  Next-hop type: flood                  Index: 289      Reference: 1
  Next-hop type: unicast                Index: 291      Reference: 3
  Next-hop interface: fe-0/1/3.0
  Next-hop type: unicast                Index: 290      Reference: 3
  Next-hop interface: fe-0/1/2.0

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: none
  Next-hop type: discard                Index: 341      Reference: 1

Destination:  fe-0/1/2.0
  Route type: dynamic
  Route reference: 0                    Route interface-index: 69
  Flags: sent to PFE
  Next-hop type: flood                  Index: 293      Reference: 1
  Next-hop type: indirect               Index: 363      Reference: 4
  Next-hop type: Push 800016
  Next-hop interface: at-1/0/1.0
  Next-hop type: indirect               Index: 301      Reference: 5
  Next hop: 10.31.3.2
  Next-hop type: Push 800000
  Next-hop interface: fe-0/1/1.0
  Next-hop type: unicast                Index: 291      Reference: 3
  Next-hop interface: fe-0/1/3.0

Destination:  fe-0/1/3.0
  Route type: dynamic
  Route reference: 0                    Route interface-index: 70
  Flags: sent to PFE
  Next-hop type: flood                  Index: 292      Reference: 1
  Next-hop type: indirect               Index: 363      Reference: 4
  Next-hop type: Push 800016
  Next-hop interface: at-1/0/1.0
  Next-hop type: indirect               Index: 301      Reference: 5
  Next hop: 10.31.3.2
  Next-hop type: Push 800000
  Next-hop interface: fe-0/1/1.0
  Next-hop type: unicast                Index: 290      Reference: 3
```

```
  Next-hop interface: fe-0/1/2.0

Destination:  00:00:5E:00:53:01/48
  Route type: dynamic
  Route reference: 0                    Route interface-index: 70
  Flags: sent to PFE, prefix load balance
  Next-hop type: unicast                Index: 291      Reference: 3
  Next-hop interface: fe-0/1/3.0
  Route used as destination:
    Packet count:         6640    Byte count:      675786
  Route used as source
    Packet count:         6894    Byte count:      696424

Destination:  00:00:5E:00:53:04/48
  Route type: dynamic
  Route reference: 0                    Route interface-index: 69
  Flags: sent to PFE, prefix load balance
  Next-hop type: unicast                Index: 290      Reference: 3
  Next-hop interface: fe-0/1/2.0
  Route used as destination:
    Packet count:           96    Byte count:        8079
  Route used as source:
    Packet count:          296    Byte count:       24955

Destination:  00:00:5E:00:53:05/48
  Route type: dynamic
  Route reference: 0                    Route interface-index: 74
  Flags: sent to PFE, prefix load balance
  Next-hop type: indirect               Index: 301      Reference: 5
  Next hop: 10.31.3.2
  Next-hop type: Push 800000
  Next-hop interface: fe-0/1/1.0
```

show route forwarding-table table default

```
user@host> show route forwarding-table table default

Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    36     2
0.0.0.0/32         perm     0                    dscd    34     1
10.0.60.0/30       user     0 10.0.60.13         ucst   713     5 fe-0/1/3.0
10.0.60.12/30      intf     0                    rslv   688     1 fe-0/1/3.0
10.0.60.12/32      dest     0 10.0.60.12         recv   686     1 fe-0/1/3.0
10.0.60.13/32      dest     0 0:5:85:8b:bc:22    ucst   713     5 fe-0/1/3.0
10.0.60.14/32      intf     0 10.0.60.14         locl   687     2
10.0.60.14/32      dest     0 10.0.60.14         locl   687     2
10.0.60.15/32      dest     0 10.0.60.15         bcst   685     1 fe-0/1/3.0
10.0.67.12/30      user     0 10.0.60.13         ucst   713     5 fe-0/1/3.0
10.0.80.0/30       ifdn     0 ff.3.0.21          ucst   676     1 so-0/0/1.0
10.0.80.0/32       dest     0 10.0.80.0          recv   678     1 so-0/0/1.0
10.0.80.2/32       user     0                    rjct    36     2
10.0.80.2/32       intf     0 10.0.80.2          locl   675     1
10.0.80.3/32       dest     0 10.0.80.3          bcst   677     1 so-0/0/1.0
10.0.90.12/30      intf     0                    rslv   684     1 fe-0/1/0.0
10.0.90.12/32      dest     0 10.0.90.12         recv   682     1 fe-0/1/0.0
10.0.90.14/32      intf     0 10.0.90.14         locl   683     2
```

```
10.0.90.14/32      dest     0 10.0.90.14         locl   683     2
10.0.90.15/32      dest     0 10.0.90.15         bcst   681     1 fe-0/1/0.0
10.5.0.0/16        user     0 192.168.187.126    ucst   324    15 fxp0.0
10.10.0.0/16       user     0 192.168.187.126    ucst   324    15 fxp0.0
10.13.10.0/23      user     0 192.168.187.126    ucst   324    15 fxp0.0
10.84.0.0/16       user     0 192.168.187.126    ucst   324    15 fxp0.0
10.150.0.0/16      user     0 192.168.187.126    ucst   324    15 fxp0.0
10.157.64.0/19     user     0 192.168.187.126    ucst   324    15 fxp0.0
10.209.0.0/16      user     0 192.168.187.126    ucst   324    15 fxp0.0

...

Routing table: default.iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    60     1

Routing table: default.inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    44     1
::/128             perm     0                    dscd    42     1
ff00::/8           perm     0                    mdsc    43     1
ff02::1/128        perm     0 ff02::1            mcst    39     1

Routing table: default.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    dscd    50     1
```

show route forwarding-table table logical-system-name/routing-instance-name

```
user@host> show route forwarding-table table R4/vpn-red
```

```
Logical system: R4
Routing table: vpn-red.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   563     1
0.0.0.0/32         perm     0                    dscd   561     2
172.16.0.1/32         user     0                    dscd   561     2
172.16.2.0/24         intf     0                    rslv   771     1 ge-1/2/0.3
172.16.2.0/32        dest     0 172.16.2.0           recv   769     1 ge-1/2/0.3
172.16.2.1/32         intf     0 172.16.2.1           locl   770     2
172.16.2.1/32         dest     0 172.16.2.1           locl   770     2
172.16.2.2/32         dest     0 0.4.80.3.0.1b.c0.d5.e4.bd.0.1b.c0.d5.e4.bc.8.0
                                                    ucst   789     1 ge-1/2/0.3
172.16.2.255/32      dest     0 172.16.2.255         bcst   768     1 ge-1/2/0.3
172.16.233.0/4        perm     1                    mdsc   562     1
172.16.233.1/32       perm     0 172.16.233.1         mcst   558     1
255.255.255.255/32 perm     0                    bcst   559     1

Logical system: R4
Routing table: vpn-red.iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   608     1
```

```
Logical system: R4
Routing table: vpn-red.inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   708     1
::/128             perm     0                    dscd   706     1
ff00::/8           perm     0                    mdsc   707     1
ff02::1/128        perm     0 ff02::1            mcst   704     1

Logical system: R4
Routing table: vpn-red.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    dscd   638
```

show route forwarding-table vpn

```
user@host> show route forwarding-table vpn VPN-A
```

```
     Routing table:: VPN-A.inet
Internet:
     Destination        Type RtRef Nexthop           Type Index NhRef Netif
     default            perm     0                   rjct     4     4
     10.39.10.20/30     intf     0 ff.3.0.21         ucst    40     1
     so-0/0/0.0
     10.39.10.21/32     intf     0 10.39.10.21       locl    36     1
     10.255.14.172/32   user     0                   ucst    69     2
     so-0/0/0.0
     10.255.14.175/32   user     0                   indr    81     3
                                                     Push 100004, Push
     100004(top) so-1/0/0.0
     172.16.233.0/4        perm     2                   mdsc     5     3
     172.16.233.1/32       perm     0 172.16.233.1         mcst     1     8
     172.16.233.5/32       user     1 172.16.233.5         mcst     1     8
     255.255.255.255/32 perm     0                   bcst     2     3
```

On QFX5200, the results for this command look like this:

```
show route forwarding-table family mpls
```

```
Routing table: default.mpls
MPLS:
Destination Type RtRef Next hop Type Index NhRef Netif
default perm 0 dscd 65 1
0 user 0 recv 64 4
1 user 0 recv 64 4
2 user 0 recv 64 4
13 user 0 recv 64 4
300384 user 0 9.1.1.1 Pop 1711 2 xe-0/0/34.0
300384(S=0) user 0 9.1.1.1 Pop 1712 2 xe-0/0/34.0
300400 user 0 ulst 131071 2
                              10.1.1.2 Pop 1713 1 xe-0/0/38.0
                              172.16.11.2 Pop 1714 1 xe-0/0/40.0
300400(S=0) user 0 ulst 131072 2
                              10.1.1.2 Pop 1715 1 xe-0/0/38.0
                              172.16.11.2 Pop 1716 1 xe-0/0/40.0
```

```
Routing table: __mpls-oam__.mpls
MPLS:
Destination Type RtRef Next hop Type Index NhRef Netif
default perm 0 dscd 1681 1
```

show route table

List of Syntax

Syntax on page 1217
Syntax (EX Series Switches, QFX Series Switches) on page 1217

Syntax

```
show route table routing-table-name
<brief | detail | extensive | terse>
<logical-system (all | logical-system-name)>
```

Syntax (EX Series Switches, QFX Series Switches)

```
show route table routing-table-name
<brief | detail | extensive | terse>
```

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 14.1X53-D15 for QFX Series switches.
Show route table evpn statement introduced in Junos OS Release 15.1X53-D30 for QFX Series switches.

Description

Display the route entries in a particular routing table.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

routing-table-name—Display route entries for all routing tables whose names begin with this string (for example, inet.0 and inet6.0 are both displayed when you run the **show route table inet** command).

Required Privilege Level

view

Related Documentation

- *show route summary*

List of Sample Output

show route table bgp.l2.vpn on page 1229
show route table bgp.l3vpn.0 on page 1229
show route table bgp.l3vpn.0 detail on page 1229
show route table bgp.rtarget.0 (When Proxy BGP Route Target Filtering Is Configured) on page 1231
show route table bgp.evpn.0 on page 1231
show route table evpna.evpn.0 on page 1231
show route table inet.0 on page 1232
show route table inet.3 on page 1232
show route table inet.3 protocol ospf on page 1232

show route table inet6.0 on page 1233
show route table inet6.3 on page 1233
show route table inetflow detail on page 1233
show route table lsdist.0 extensive on page 1234
show route table l2circuit.0 on page 1236
show route table mpls on page 1236
show route table mpls extensive on page 1236
show route table mpls.0 on page 1237
show route table mpls.0 detail (PTX Series) on page 1238
show route table mpls.0 ccc ge-0/0/1.1004 detail on page 1238
show route table mpls.0 protocol evpn on page 1240
show route table mpls.0 protocol ospf on page 1246
show route table mpls.0 extensive (PTX Series) on page 1246
show route table mpls.0 (RSVP Route—Transit LSP) on page 1247
show route table vpls_1 detail on page 1247
show route table vpn-a on page 1248
show route table vpn-a.mdt.0 on page 1248
show route table VPN-A detail on page 1248
show route table VPN-AB.inet.0 on page 1249
show route table VPN_blue.mvpn-inet6.0 on page 1249
show route table vrf1.mvpn.0 extensive on page 1250
show route table inetflow detail on page 1250
show route table bgp.evpn.0 extensive |no-more (EVPN) on page 1254

Output Fields Table 43 on page 1218 describes the output fields for the **show route table** command. Output fields are listed in the approximate order in which they appear.

Table 43: show route table Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).

Table 43: show route table Output Fields (continued)

Field Name	Field Description

Restart complete	All protocols have restarted for this routing table.
	
	Restart state:
	
	• **Pending:*protocol-name***—List of protocols that have not yet completed graceful restart for this routing table.
	• **Complete**—All protocols have restarted for this routing table.
	
	For example, if the output shows-
	
	• `LDP.inet.0 : 5 routes (4 active, 1 holddown, 0 hidden)`
	`Restart Pending: OSPF LDP VPN`
	
	This indicates that **OSPF**, **LDP**, and **VPN** protocols did not restart for the **LDP.inet.0** routing table.
	
	• `vpls_1.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)`
	`Restart Complete`
	
	This indicates that all protocols have restarted for the **vpls_1.l2vpn.0** routing table.

number destinations	Number of destinations for which there are routes in the routing table.
number routes	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active) • **holddown** (routes that are in the pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)

Table 43: show route table Output Fields (continued)

Field Name	Field Description

route-destination (entry, announced)	Route destination (for example:10.0.0.1/24). The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination. Sometimes the route destination is presented in another format, such as:
	• ***MPLS-label*** (for example, 80001).
	• ***interface-name*** (for example, ge-1/0/2).
	• ***neighbor-address*:*control-word-status*:*encapsulation type*:*vc-id*:*source*** (Layer 2 circuit only; for example, 10.1.1.195:NoCtrlWord:1:1:Local/96).
	  • ***neighbor-address***—Address of the neighbor.
	  • ***control-word-status***—Whether the use of the control word has been negotiated for this virtual circuit: **NoCtrlWord** or **CtrlWord**.
	  • ***encapsulation type***—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport.
	  • ***vc-id***—Virtual circuit identifier.
	  • ***source***—Source of the advertisement: **Local** or **Remote**.
	• ***inclusive multicast Ethernet tag route***—Type of route destination represented by (for example, 3:100.100.100.10:100::0::10::100.100.100.10/384):
	  • *route distinguisher*—(8 octets) Route distinguisher (RD) must be the RD of the EVPN instance (EVI) that is advertising the NLRI.
	  • *Ethernet tag ID*—(4 octets) Identifier of the Ethernet tag. Can set to 0 or to a valid Ethernet tag value.
	  • *IP address length*—(1 octet) Length of IP address in bits.
	  • *originating router's IP address*—(4 or 16 octets) Must set to the provider edge (PE) device's IP address. This address should be common for all EVIs on the PE device, and may be the PE device's loopback address.

label stacking	**(**Next-to-the-last-hop routing device for MPLS only) Depth of the MPLS label stack, where the label-popping operation is needed to remove one or more labels from the top of the stack. A pair of routes is displayed, because the pop operation is performed only when the stack depth is two or more labels. • **S=0 route** indicates that a packet with an incoming label stack depth of 2 or more exits this routing device with one fewer label (the label-popping operation is performed). • If there is no **S=** information, the route is a normal MPLS route, which has a stack depth of 1 (the label-popping operation is not performed).
[*protocol*, *preference*]	Protocol from which the route was learned and the preference value for the route. • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.

Table 43: show route table Output Fields (continued)

Field Name	Field Description
Level	(IS-IS only). In IS-IS, a single AS can be divided into smaller groups called areas. Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas. This organization is accomplished by configuring Level 1 and Level 2 intermediate systems. Level 1 systems route within an area. When the destination is outside an area, they route toward a Level 2 system. Level 2 intermediate systems route between areas and toward other ASs.
Route Distinguisher	IP subnet augmented with a 64-bit prefix.
PMSI	Provider multicast service interface (MVPN routing table).
Next-hop type	Type of next hop. For a description of possible values for this field, see Table 44 on page 1224.
Next-hop reference count	Number of references made to the next hop.
Flood nexthop branches exceed maximum message	Indicates that the number of flood next-hop branches exceeded the system limit of 32 branches, and only a subset of the flood next-hop branches were installed in the kernel.
Source	IP address of the route source.
Next hop	Network layer address of the directly reachable neighboring system.
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the name of the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • Weight—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • Balance—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.
Label-switched-path *lsp-path-name*	Name of the LSP used to reach the next hop.
Label operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).
Interface	(Local only) Local interface name.
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to derive a forwarding next hop.
Indirect next hop	Index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.
State	State of the route (a route can be in more than one state). See Table 45 on page 1226.

Table 43: show route table Output Fields (continued)

Field Name	Field Description
Local AS	AS number of the local routing devices.
Age	How long the route has been known.
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.
Metric*n*	Cost value of the indicated route. For routes within an AS, the cost is determined by IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.
MED-plus-IGP	Metric value for BGP path selection to which the IGP cost to the next-hop destination has been added.
TTL-Action	For MPLS LSPs, state of the TTL propagation attribute. Can be enabled or disabled for all RSVP-signaled and LDP-signaled LSPs or for specific VRF routing instances.
Task	Name of the protocol that has added the route.
Announcement bits	The number of BGP peers or protocols to which Junos OS has announced this route, followed by the list of the recipients of the announcement. Junos OS can also announce the route to the kernel routing table (KRT) for installing the route into the Packet Forwarding Engine, to a resolve tree, a Layer 2 VC, or even a VPN. For example, ***n*-Resolve inet** indicates that the specified route is used for route resolution for next hops found in the routing table. • ***n***—An index used by Juniper Networks customer support only.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **Recorded**—The AS path is recorded by the sample process (sampled). • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the number that precedes the AS path. This number represents the number of ASs present in the AS path, when calculated as defined in RFC 4271. This value is used in the AS-path merge process, as defined in RFC 4893. • **[ ]**—If more than one AS number is configured on the routing device, or if AS path prepending is configured, brackets enclose the local AS number associated with the AS path. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.

Table 43: show route table Output Fields (continued)

Field Name	Field Description
validation-state	(BGP-learned routes) Validation status of the route: • **Invalid**—Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database. • **Unknown**—Indicates that the prefix is not among the prefixes or prefix ranges in the database. • **Unverified**—Indicates that the origin of the prefix is not verified against the database. This is because the database got populated and the validation is not called for in the BGP import policy, although origin validation is enabled, or the origin validation is not enabled for the BGP peers. • **Valid**—Indicates that the prefix and autonomous system pair are found in the database.
FECs bound to route	Indicates point-to-multipoint root address, multicast source address, and multicast group address when multipoint LDP (M-LDP) inband signaling is configured.
Primary Upstream	When multipoint LDP with multicast-only fast reroute (MoFRR) is configured, indicates the primary upstream path. MoFRR transmits a multicast join message from a receiver toward a source on a primary path, while also transmitting a secondary multicast join message from the receiver toward the source on a backup path.
RPF Nexthops	When multipoint LDP with MoFRR is configured, indicates the reverse-path forwarding (RPF) next-hop information. Data packets are received from both the primary path and the secondary paths. The redundant packets are discarded at topology merge points due to the RPF checks.
Label	Multiple MPLS labels are used to control MoFRR stream selection. Each label represents a separate route, but each references the same interface list check. Only the primary label is forwarded while all others are dropped. Multiple interfaces can receive packets using the same label.
weight	Value used to distinguish MoFRR primary and backup routes. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible.
VC Label	MPLS label assigned to the Layer 2 circuit virtual connection.
MTU	Maximum transmission unit (MTU) of the Layer 2 circuit.
VLAN ID	VLAN identifier of the Layer 2 circuit.
Prefixes bound to route	Forwarding equivalent class (FEC) bound to this route. Applicable only to routes installed by LDP.
Communities	Community path attribute for the route. See Table 46 on page 1228 for all possible values for this field.
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).
control flags	Control flags: **none** or **Site Down**.
mtu	Maximum transmission unit (MTU) information.
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.
status vector	Layer 2 VPN and VPLS network layer reachability information (NLRI).

Table 43: show route table Output Fields (continued)

Field Name	Field Description
Accepted Multipath	Current active path when BGP multipath is configured.
Accepted LongLivedStale	The LongLivedStale flag indicates that the route was marked LLGR-stale by this router, as part of the operation of LLGR receiver mode. Either this flag or the LongLivedStaleImport flag might be displayed for a route. Neither of these flags is displayed at the same time as the Stale (ordinary GR stale) flag.
Accepted LongLivedStaleImport	The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy. Either this flag or the LongLivedStale flag might be displayed for a route. Neither of these flags is displayed at the same time as the Stale (ordinary GR stale) flag. Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and import into the inet.0 routing table
ImportAccepted LongLivedStaleImport	Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and imported into the inet.0 routing table The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy.
Accepted MultipathContrib	Path currently contributing to BGP multipath.
Localpref	Local preference value included in the route.
Router ID	BGP router ID as advertised by the neighbor in the open message.
Primary Routing Table	In a routing table group, the name of the primary routing table in which the route resides.
Secondary Tables	In a routing table group, the name of one or more secondary tables in which the route resides.

Table 44 on page 1224 describes all possible values for the Next-hop Types output field.

Table 44: Next-hop Types Output Field Values

Next-Hop Type	Description
Broadcast (bcast)	Broadcast next hop.
Deny	Deny next hop.
Discard	Discard next hop.
Flood	Flood next hop. Consists of components called branches, up to a maximum of 32 branches. Each flood next-hop branch sends a copy of the traffic to the forwarding interface. Used by point-to-multipoint RSVP, point-to-multipoint LDP, point-to-multipoint CCC, and multicast.

Table 44: Next-hop Types Output Field Values (continued)

Next-Hop Type	Description
Hold	Next hop is waiting to be resolved into a unicast or multicast type.
Indexed (idxd)	Indexed next hop.
Indirect (indr)	Used with applications that have a protocol next hop address that is remote. You are likely to see this next-hop type for internal BGP (IBGP) routes when the BGP next hop is a BGP neighbor that is not directly connected.
Interface	Used for a network address assigned to an interface. Unlike the router next hop, the interface next hop does not reference any specific node on the network.
Local (locl)	Local address on an interface. This next-hop type causes packets with this destination address to be received locally.
Multicast (mcst)	Wire multicast next hop (limited to the LAN).
Multicast discard (mdsc)	Multicast discard.
Multicast group (mgrp)	Multicast group member.
Receive (recv)	Receive.
Reject (rjct)	Discard. An ICMP unreachable message was sent.
Resolve (rslv)	Resolving next hop.
Routed multicast (mcrt)	Regular multicast next hop.
Router	A specific node or set of nodes to which the routing device forwards packets that match the route prefix. To qualify as a next-hop type router, the route must meet the following criteria: • Must not be a direct or local subnet for the routing device. • Must have a next hop that is directly connected to the routing device.
Table	Routing table next hop.
Unicast (ucst)	Unicast.
Unilist (ulst)	List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.

Table 45 on page 1226 describes all possible values for the State output field. A route can be in more than one state (for example, **<Active NoReadvrt Int Ext>**).

Table 45: State Output Field Values

Value	Description
Accounting	Route needs accounting.
Active	Route is active.
Always Compare MED	Path with a lower multiple exit discriminator (MED) is available.
AS path	Shorter AS path is available.
Cisco Non-deterministic MED selection	Cisco nondeterministic MED is enabled, and a path with a lower MED is available.
Clone	Route is a clone.
Cluster list length	Length of cluster list sent by the route reflector.
Delete	Route has been deleted.
Ex	Exterior route.
Ext	BGP route received from an external BGP neighbor.
FlashAll	Forces all protocols to be notified of a change to any route, active or inactive, for a prefix. When not set, protocols are informed of a prefix only when the active route changes.
Hidden	Route not used because of routing policy.
IfCheck	Route needs forwarding RPF check.
IGP metric	Path through next hop with lower IGP metric is available.
Inactive reason	Flags for this route, which was not selected as best for a particular destination.
Initial	Route being added.
Int	Interior route.
Int Ext	BGP route received from an internal BGP peer or a BGP confederation peer.
Interior > Exterior > Exterior via Interior	Direct, static, IGP, or EBGP path is available.

Table 45: State Output Field Values (continued)

Value	Description
Local Preference	Path with a higher local preference value is available.
Martian	Route is a martian (ignored because it is obviously invalid).
MartianOK	Route exempt from martian filtering.
Next hop address	Path with lower metric next hop is available.
No difference	Path from neighbor with lower IP address is available.
NoReadvrt	Route not to be advertised.
NotBest	Route not chosen because it does not have the lowest MED.
Not Best in its group	Incoming BGP AS is not the best of a group (only one AS can be the best).
NotInstall	Route not to be installed in the forwarding table.
Number of gateways	Path with a greater number of next hops is available.
Origin	Path with a lower origin code is available.
Pending	Route pending because of a hold-down configured on another route.
Release	Route scheduled for release.
RIB preference	Route from a higher-numbered routing table is available.
Route Distinguisher	64-bit prefix added to IP subnets to make them unique.
Route Metric or MED comparison	Route with a lower metric or MED is available.
Route Preference	Route with lower preference value is available.
Router ID	Path through a neighbor with lower ID is available.
Secondary	Route not a primary route.
Unusable path	Path is not usable because of one of the following conditions: • The route is damped. • The route is rejected by an import policy. • The route is unresolved.
Update source	Last tiebreaker is the lowest IP address value.

Table 46 on page 1228 describes the possible values for the Communities output field.

Table 46: Communities Output Field Values

Value	Description
area-number	4 bytes, encoding a 32-bit area number. For AS-external routes, the value is **0**. A nonzero value identifies the route as internal to the OSPF domain, and as within the identified area. Area numbers are relative to a particular OSPF domain.
bandwidth: ***local AS number:link-bandwidth-number***	Link-bandwidth community value used for unequal-cost load balancing. When BGP has several candidate paths available for multipath purposes, it does not perform unequal-cost load balancing according to the link-bandwidth community unless all candidate paths have this attribute.
domain-id	Unique configurable number that identifies the OSPF domain.
domain-id-vendor	Unique configurable number that further identifies the OSPF domain.
link-bandwidth-number	Link-bandwidth number: from **0** through **4,294,967,295** (bytes per second).
local AS number	Local AS number: from **1** through **65,535**.
options	1 byte. Currently this is only used if the route type is **5** or **7**. Setting the least significant bit in the field indicates that the route carries a type 2 metric.
origin	(Used with VPNs) Identifies where the route came from.
ospf-route-type	1 byte, encoded as **1** or **2** for intra-area routes (depending on whether the route came from a type 1 or a type 2 LSA); **3** for summary routes; **5** for external routes (area number must be**0**); **7** for NSSA routes; or **129** for sham link endpoint addresses.
route-type-vendor	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x8000**. The format is ***area-number:ospf-route-type:options***.
rte-type	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x0306**. The format is ***area-number:ospf-route-type:options***.
target	Defines which VPN the route participates in; **target** has the format ***32-bit IP address:16-bit number***. For example, 10.19.0.0:100.
unknown IANA	Incoming IANA codes with a value between **0x1** and **0x7fff**. This code of the BGP extended community attribute is accepted, but it is not recognized.
unknown OSPF vendor community	Incoming IANA codes with a value above **0x8000**. This code of the BGP extended community attribute is accepted, but it is not recognized.

Sample Output

show route table bgp.l2.vpn

```
user@host> show route table bgp.l2.vpn

bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.24.1:1:4:1/96
                   *[BGP/170] 01:08:58, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
```

show route table bgp.l3vpn.0

```
user@host> show route table bgp.l3vpn.0

     bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
     + = Active Route, - = Last Active, * = Both

     10.255.71.15:100:10.255.71.17/32
                        *[BGP/170] 00:03:59, MED 1, localpref 100, from
     10.255.71.15
                           AS path: I
                         > via so-2/1/0.0, Push 100020, Push 100011(top)
     10.255.71.15:200:10.255.71.18/32
                        *[BGP/170] 00:03:59, MED 1, localpref 100, from
     10.255.71.15
                           AS path: I
                         > via so-2/1/0.0, Push 100021, Push 100011(top)
```

show route table bgp.l3vpn.0 detail

```
user@host> show route table bgp.l3vpn.0 detail

bgp.l3vpn.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)

10.255.245.12:1:172.16.4.0/8 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
         Route Distinguisher: 10.255.245.12:1
         Source: 10.255.245.12
         Next hop: 192.168.208.66 via fe-0/0/0.0, selected
         Label operation: Push 182449
         Protocol next hop: 10.255.245.12
         Push 182449
         Indirect next hop: 863a630 297
         State: <Active Int Ext>
         Local AS:    35 Peer AS:    35
         Age: 12:19      Metric2: 1
         Task: BGP_35.10.255.245.12+179
         Announcement bits (1): 0-BGP.0.0.0.0+179
         AS path: 30 10458 14203 2914 3356 I (Atomic) Aggregator: 3356 4.68.0.11

         Communities: 2914:420 target:11111:1 origin:56:78
         VPN Label: 182449
```

```
        Localpref: 100
         Router ID: 10.255.245.12

10.255.245.12:1:4.17.225.0/24 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
        Route Distinguisher: 10.255.245.12:1
        Source: 10.255.245.12
        Next hop: 192.168.208.66 via fe-0/0/0.0, selected
        Label operation: Push 182465
        Protocol next hop: 10.255.245.12
        Push 182465
        Indirect next hop: 863a8f0 305
        State: <Active Int Ext>
        Local AS:    35 Peer AS:    35
        Age: 12:19      Metric2: 1
        Task: BGP_35.10.255.245.12+179
        Announcement bits (1): 0-BGP.0.0.0.0+179
AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496 6496 I
               Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
               VPN Label: 182465
               Localpref: 100
               Router ID: 10.255.245.12

10.255.245.12:1:4.17.226.0/23 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
        Route Distinguisher: 10.255.245.12:1
        Source: 10.255.245.12
        Next hop: 192.168.208.66 via fe-0/0/0.0, selected
        Label operation: Push 182465
        Protocol next hop: 10.255.245.12
        Push 182465
        Indirect next hop: 86bd210 330
        State: <Active Int Ext>
        Local AS:    35 Peer AS:    35
        Age: 12:19      Metric2: 1
        Task: BGP_35.10.255.245.12+179
        Announcement bits (1): 0-BGP.0.0.0.0+179
        AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496

        6496 I
        Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
        VPN Label: 182465
        Localpref: 100
        Router ID: 10.255.245.12

10.255.245.12:1:4.17.251.0/24 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
        Route Distinguisher: 10.255.245.12:1
        Source: 10.255.245.12
        Next hop: 192.168.208.66 via fe-0/0/0.0, selected
        Label operation: Push 182465
        Protocol next hop: 10.255.245.12
        Push 182465
        Indirect next hop: 86bd210 330
        State: <Active Int Ext>
        Local AS:    35 Peer AS:    35
        Age: 12:19      Metric2: 1
        Task: BGP_35.10.255.245.12+179
        Announcement bits (1): 0-BGP.0.0.0.0+179
```

```
        AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496

        6496 I
        Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
        VPN Label: 182465
        Localpref: 100
```

show route table bgp.rtarget.0 (When Proxy BGP Route Target Filtering Is Configured)

```
user@host> show route table bgp.rtarget.o

bgp.rtarget.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

100:100:100/96
                   *[RTarget/5] 00:03:14
                      Type Proxy
                        for 10.255.165.103
                        for 10.255.166.124
                      Local
```

show route table bgp.evpn.0

```
user@host> show route table bgp.evpn.0

bgp.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2:100.100.100.2:100::0::00:26:88:5f:67:b0/304
                   *[BGP/170] 11:00:05, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
2:100.100.100.2:100::0::00:51:51:51:51:51/304
                   *[BGP/170] 11:00:05, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
2:100.100.100.3:100::0::00:52:52:52:52:52/304
                   *[BGP/170] 10:59:58, localpref 100, from 100.100.100.3
                      AS path: I, validation-state: unverified
                    > to 100.64.13.3 via ge-2/0/8.0, label-switched-path R0toR2
2:100.100.100.3:100::0::a8:d0:e5:5b:01:c8/304
                   *[BGP/170] 10:59:58, localpref 100, from 100.100.100.3
                      AS path: I, validation-state: unverified
                    > to 100.64.13.3 via ge-2/0/8.0, label-switched-path R0toR2
3:100.100.100.2:100::1000::100.100.100.2/304
                   *[BGP/170] 11:00:16, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
3:100.100.100.2:100::2000::100.100.100.2/304
                   *[BGP/170] 11:00:16, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
```

show route table evpna.evpn.0

```
user@host> show route table evpna.evpn.0
```

```
evpna.evpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

3:100.100.100.10:100::0::10::100.100.100.10/384
                   *[EVPN/170] 01:37:09
                      Indirect
3:100.100.100.2:100::2000::100.100.100.2/304
                   *[EVPN/170] 01:37:12
                      Indirect
```

show route table inet.0

user@host> **show route table inet.0**

```
inet.0: 12 destinations, 12 routes (11 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:51:57
                    > to 172.16.5.254 via fxp0.0
10.0.0.1/32         *[Direct/0] 00:51:58
                    > via at-5/3/0.0
10.0.0.2/32         *[Local/0] 00:51:58
                     Local
10.12.12.21/32     *[Local/0] 00:51:57
                     Reject
10.13.13.13/32     *[Direct/0] 00:51:58
                    > via t3-5/2/1.0
10.13.13.14/32     *[Local/0] 00:51:58
                     Local
10.13.13.21/32     *[Local/0] 00:51:58
                     Local
10.13.13.22/32     *[Direct/0] 00:33:59
                    > via t3-5/2/0.0
127.0.0.1/32        [Direct/0] 00:51:58
                    > via lo0.0
10.222.5.0/24     *[Direct/0] 00:51:58
                    > via fxp0.0
10.222.5.81/32    *[Local/0] 00:51:58
                      Local
```

show route table inet.3

user@host> **show route table inet.3**

```
inet.3: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32        *[LDP/9] 00:25:43, metric 10, tag 200
                      to 10.2.94.2 via lt-1/2/0.49
                    > to 10.2.3.2 via lt-1/2/0.23
```

show route table inet.3 protocol ospf

user@host> **show route table inet.3 protocol ospf**

```
inet.3: 9 destinations, 18 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.20/32         [L-OSPF/10] 1d 00:00:56, metric 2
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800020
                     to 10.0.6.60 via lt-1/2/0.12, Push 800020, Push 800030(top)
1.1.1.30/32         [L-OSPF/10] 1d 00:01:01, metric 3
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800030
                      to 10.0.6.60 via lt-1/2/0.12, Push 800030
1.1.1.40/32         [L-OSPF/10] 1d 00:01:01, metric 4
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800040
                      to 10.0.6.60 via lt-1/2/0.12, Push 800040
1.1.1.50/32         [L-OSPF/10] 1d 00:01:01, metric 5
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800050
                      to 10.0.6.60 via lt-1/2/0.12, Push 800050
1.1.1.60/32         [L-OSPF/10] 1d 00:01:01, metric 6
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800060
                      to 10.0.6.60 via lt-1/2/0.12, Pop
```

show route table inet6.0

```
user@host> show route table inet6.0

inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Route, * = Both

fec0:0:0:3::/64 *[Direct/0] 00:01:34
>via fe-0/1/0.0

fec0:0:0:3::/128 *[Local/0] 00:01:34
>Local

fec0:0:0:4::/64 *[Static/5] 00:01:34
>to fec0:0:0:3::ffff via fe-0/1/0.0
```

show route table inet6.3

```
user@router> show route table inet6.3

inet6.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

::10.255.245.195/128
                   *[LDP/9] 00:00:22, metric 1
                    > via so-1/0/0.0
::10.255.245.196/128
                   *[LDP/9] 00:00:08, metric 1
                    > via so-1/0/0.0, Push 100008
```

show route table inetflow detail

```
user@host> show route table inetflow detail

inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.12.44.1,*/48 (1 entry, 1 announced)
```

```
       *BGP    Preference: 170/-101
               Next-hop reference count: 2
               State: <Active Ext>
               Local AS: 64502 Peer AS: 64500
               Age: 4
               Task: BGP_64500.10.12.99.5+3792
               Announcement bits (1): 0-Flow
               AS path: 64500 I
               Communities: traffic-rate:0:0
               Validation state: Accept, Originator: 10.12.99.5
               Via: 10.12.44.0/24, Active
               Localpref: 100
               Router ID: 10.255.71.161

10.12.56.1,*/48 (1 entry, 1 announced)
       *Flow   Preference: 5
               Next-hop reference count: 2
               State: <Active>
               Local AS: 64502
               Age: 6:30
               Task: RT Flow
               Announcement bits (2): 0-Flow 1-BGP.0.0.0.0+179
               AS path: I
               Communities: 1:1
```

show route table lsdist.0 extensive

```
user@host> show route table lsdist.0 extensive

lsdist.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
NODE { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 ISIS-L1:0 }/1152
 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group ibgp type Internal) Type 1 val 0xa62f378 (adv_entry)
   Advertised metrics:
     Nexthop: Self
     Localpref: 100
     AS path: [4170512532] I
     Communities:
Path NODE { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 ISIS-L1:0 }
 Vector len 4.  Val: 0
       *IS-IS  Preference: 15
               Level: 1
               Next hop type: Fictitious, Next hop index: 0
               Address: 0x95dfc64
               Next-hop reference count: 9
               State: <Active NotInstall>
               Local AS: 4170512532
               Age: 6:05
               Validation State: unverified
               Task: IS-IS
               Announcement bits (1): 0-BGP_RT_Background
               AS path: I
               IPv4 Router-ids:
                  128.220.11.197
               Area membership:
                  47 00 05 80 ff f8 00 00 00 01 08 00 01
              SPRING-Capabilities:                 - SRGB block [Start: 800000,
```

```
 Range: 256, Flags: 0xc0]
                SPRING-Algorithms:
                   - Algo: 0
 LINK { Local { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 }.{
IPv4:8.65.1.105 } Remote { AS:4170512532 BGP-LS ID:4170512532 ISO:4284.3300.5067)
TSI:
Page 0 idx 0, (group ibgp type Internal) Type 1 val 0xa62f3cc (adv_entry)
    Advertised metrics:
      Nexthop: Self
      Localpref: 100
      AS path: [4170512532] I
      Communities:
Path LINK { Local { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 }.{
 IPv4:8.65.1.105 } Remote { AS:4170512532 BGP-LS ID:4170512532 ISO:4284.33000
      *IS-IS  Preference: 15
                Level: 1
                Next hop type: Fictitious, Next hop index: 0
                Address: 0x95dfc64
                Next-hop reference count: 9
                State: <Active NotInstall>
                Local AS: 4170512532
                Age: 6:05
                Validation State: unverified
                Task: IS-IS
                Announcement bits (1): 0-BGP_RT_Background
                AS path: I
                Color: 32768
                Maximum bandwidth: 1000Mbps
                Reservable bandwidth: 1000Mbps
                Unreserved bandwidth by priority:
                   0   1000Mbps
                   1   1000Mbps
                   2   1000Mbps
                   3   1000Mbps
                   4   1000Mbps
                   5   1000Mbps
                   6   1000Mbps
                   7   1000Mbps
                 Metric: 10
                 TE Metric: 10
                 LAN IPV4 Adj-SID -  Label: 299776, Flags: 0x30,
Weight: 0, Nbr: 10.220.1.83

PREFIX { Node { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 } {
IPv4:128.220.11.197/32 } ISIS-L1:0 }/1152 (1 entry, 1 announced) TSI: Page 0 idx
 0, (group ibgp type Internal) Type 1 val 0xa62f43c (adv_entry)
    Advertised metrics:
      Nexthop: Self
      Localpref: 100
      AS path: [4170512532] I
      Communities:
Path PREFIX { Node { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 }
{ IPv4:128.220.11.197/32 } ISIS-L1:0 } Vector len 4.  Val: 0
         *IS-IS  Preference: 15
                 Level: 1
                 Next hop type: Fictitious, Next hop index: 0
                 Address: 0x95dfc64
                 Next-hop reference count: 9
                 State:<Active NotInstall>
```

```
                Local AS: 4170512532
                Age: 6:05
                Validation State: unverified
                Task: IS-IS
                Announcement bits (1): 0-BGP_RT_Background
                AS path: I
                             Prefix SID: 67, Flags: 0x40, Algo: 0
```

show route table l2circuit.0

```
user@host> show route table l2circuit.0

l2circuit.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.195:NoCtrlWord:1:1:Local/96
                   *[L2CKT/7] 00:50:47
                    > via so-0/1/2.0, Push 100049
                      via so-0/1/3.0, Push 100049
10.1.1.195:NoCtrlWord:1:1:Remote/96
                   *[LDP/9] 00:50:14
                      Discard
10.1.1.195:CtrlWord:1:2:Local/96
                   *[L2CKT/7] 00:50:47
                    > via so-0/1/2.0, Push 100049
                      via so-0/1/3.0, Push 100049
10.1.1.195:CtrlWord:1:2:Remote/96
                   *[LDP/9] 00:50:14
                      Discard
```

show route table mpls

```
user@host> show route table mpls

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 00:13:55, metric 1
                      Receive
1                  *[MPLS/0] 00:13:55, metric 1
                      Receive
2                  *[MPLS/0] 00:13:55, metric 1
                      Receive
1024               *[VPN/0] 00:04:18
                      to table red.inet.0, Pop
```

show route table mpls extensive

```
user@host> show route table mpls extensive

100000 (1 entry, 1 announced)
TSI:
KRT in-kernel 100000 /36 -> {so-1/0/0.0}
        *LDP    Preference:   9
                Next hop:  via so-1/0/0.0, selected
```

```
                Pop
                State: <Active Int>
                Age: 29:50      Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
                Prefixes bound to route: 10.0.0.194/32
```

show route table mpls.0

```
user@host> show route table mpls.0

mpls.0: 18 destinations, 19 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 11:39:56, metric 1
                      to table inet.0
0(S=0)             *[MPLS/0] 11:39:56, metric 1
                      to table mpls.0
1                  *[MPLS/0] 11:39:56, metric 1
                      Receive
2                  *[MPLS/0] 11:39:56, metric 1
                      to table inet6.0
2(S=0)             *[MPLS/0] 11:39:56, metric 1
                      to table mpls.0
13                 *[MPLS/0] 11:39:56, metric 1
                      Receive
303168             *[EVPN/7] 11:00:49, routing-instance pbbn10, route-type
Ingress-MAC, ISID 0
                      to table pbbn10.evpn-mac.0
303184             *[EVPN/7] 11:00:53, routing-instance pbbn10, route-type
Ingress-IM, ISID 1000
                      to table pbbn10.evpn-mac.0
                    [EVPN/7] 11:00:53, routing-instance pbbn10, route-type
Ingress-IM, ISID 2000
                      to table pbbn10.evpn-mac.0
303264             *[EVPN/7] 11:00:53, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-IM, ISID 1000
                     > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303280             *[EVPN/7] 11:00:53, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-IM, ISID 2000
                     > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303328             *[EVPN/7] 11:00:49, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-MAC, ISID 0
                     > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303344             *[EVPN/7] 11:00:49, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-MAC, ISID 0
                     > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303360             *[EVPN/7] 11:00:47, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC 00:26:88:5f:67:b0
                     > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303376             *[EVPN/7] 11:00:47, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC 00:51:51:51:51:51
                     > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303392             *[EVPN/7] 11:00:35, remote-pe 100.100.100.3, routing-instance
 pbbn10, route-type Egress-MAC, ISID 0
                     > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
303408             *[EVPN/7] 11:00:35, remote-pe 100.100.100.3, routing-instance
```

```
 pbbn10, route-type Egress-MAC, ISID 0
                    > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
303424             *[EVPN/7] 11:00:33, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC a8:d0:e5:5b:01:c8
                    > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
303440             *[EVPN/7] 11:00:33, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC 00:52:52:52:52:52
                    > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
```

show route table mpls.0 detail (PTX Series)

```
user@host> show route table mpls.0 detail

ge-0/0/2.600 (1 entry, 1 announced)
        *L2VPN  Preference: 7
                Next hop type: Indirect
                Address: 0x9438f34
                Next-hop reference count: 2
                Next hop type: Router, Next hop index: 567
                Next hop: 10.0.0.1 via ge-0/0/1.0, selected
                Label operation: Push 299808
                Label TTL action: prop-ttl
                Load balance label: Label 299808:None;
                Session Id: 0x1
                Protocol next hop: 10.255.255.1
                Label operation: Push 299872 Offset: 252
                Label TTL action: no-prop-ttl
                Load balance label: Label 299872:Flow label PUSH;
                Composite next hop: 0x9438ed8 570 INH Session ID: 0x2
                Indirect next hop: 0x9448208 262142 INH Session ID: 0x2
                State: <Active Int>
                Age: 21         Metric2: 1
                Validation State: unverified
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 2-Common L2 VC
                AS path: I
```

show route table mpls.0 ccc ge-0/0/1.1004 detail

```
user@host>show route table mpls.0 ccc ge-0/0/1.1004 detail

mpls.0: 121 destinations, 121 routes (121 active, 0 holddown, 0 hidden)
ge-0/0/1.1004 (1 entry, 1 announced)
        *EVPN   Preference: 7
                Next hop type: List, Next hop index: 1048577
                Address: 0xdc14770
                Next-hop reference count: 3
                Next hop: ELNH Address 0xd011e30
                    Next hop type: Indirect, Next hop index: 0
                    Address: 0xd011e30
                    Next-hop reference count: 3
                    Protocol next hop: 100.100.100.1
                    Label operation: Push 301952
                    Composite next hop: 0xd011dc0 754 INH Session ID: 0x146
                    Indirect next hop: 0xb69a890 1048615 INH Session ID: 0x146
                        Next hop type: Router, Next hop index: 735
```

```
                Address: 0xd00e530
                Next-hop reference count: 23
                Next hop: 100.46.1.2 via ge-0/0/5.0
                Label-switched-path pe4_to_pe1
                Label operation: Push 300320
                Label TTL action: prop-ttl
                Load balance label: Label 300320: None;
                Label element ptr: 0xd00e580
                Label parent element ptr: 0x0
                Label element references: 18
                Label element child references: 16
                Label element lsp id: 5
        Next hop: ELNH Address 0xd012070
            Next hop type: Indirect, Next hop index: 0
            Address: 0xd012070
            Next-hop reference count: 3
            Protocol next hop: 100.100.100.2
            Label operation: Push 301888
            Composite next hop: 0xd012000 755 INH Session ID: 0x143
            Indirect next hop: 0xb69a9a0 1048641 INH Session ID: 0x143
                Next hop type: Router, Next hop index: 716
                Address: 0xd00e710
                Next-hop reference count: 23
                Next hop: 100.46.1.2 via ge-0/0/5.0
                Label-switched-path pe4_to_pe2
                Label operation: Push 300304
                Label TTL action: prop-ttl
                Load balance label: Label 300304: None;
                Label element ptr: 0xd00e760
                Label parent element ptr: 0x0
                Label element references: 15
                Label element child references: 13
                Label element lsp id: 6
        Next hop: ELNH Address 0xd0121f0, selected
            Next hop type: Indirect, Next hop index: 0
            Address: 0xd0121f0
            Next-hop reference count: 3
            Protocol next hop: 100.100.100.3
            Label operation: Push 301984
            Composite next hop: 0xd012180 756 INH Session ID: 0x145
            Indirect next hop: 0xb69aab0 1048642 INH Session ID: 0x145
                Next hop type: Router, Next hop index: 801
                Address: 0xd010ed0
                Next-hop reference count: 32
                Next hop: 100.46.1.2 via ge-0/0/5.0
                Label-switched-path pe4_to_pe3
                Label operation: Push 300336
                Label TTL action: prop-ttl
                Load balance label: Label 300336: None;
                Label element ptr: 0xd0108c0
                Label parent element ptr: 0x0
                Label element references: 22
                Label element child references: 20
                Label element lsp id: 7
        State: < Active Int >
        Age: 2:06:50
        Validation State: unverified
        Task: evpn global task
```

```
                Announcement bits (1): 1-KRT
                AS path: I
```

show route table mpls.0 protocol evpn

user@host>**show route table mpls.0 protocol evpn**

```
mpls.0: 121 destinations, 121 routes (121 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

299872             *[EVPN/7] 02:30:58, routing-instance mhevpn, route-type
Ingress-IM, vlan-id 10
                      to table mhevpn.evpn-mac.0
300016             *[EVPN/7] 02:30:38, routing-instance VS-1, route-type
Ingress-IM, vlan-id 110
                      to table VS-1.evpn-mac.0
300032             *[EVPN/7] 02:30:38, routing-instance VS-1, route-type
Ingress-IM, vlan-id 120
                      to table VS-1.evpn-mac.0
300048             *[EVPN/7] 02:30:38, routing-instance VS-1, route-type
Ingress-IM, vlan-id 130
                      to table VS-1.evpn-mac.0
300064             *[EVPN/7] 02:30:38, routing-instance VS-2, route-type
Ingress-IM, vlan-id 210
                      to table VS-2.evpn-mac.0
300080             *[EVPN/7] 02:30:38, routing-instance VS-2, route-type
Ingress-IM, vlan-id 220
                      to table VS-2.evpn-mac.0
300096             *[EVPN/7] 02:30:38, routing-instance VS-2, route-type
Ingress-IM, vlan-id 230
                      to table VS-2.evpn-mac.0
300112             *[EVPN/7] 02:27:06, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:44:44:44:44:44:44:44:44:44
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300128             *[EVPN/7] 02:29:22, routing-instance mhevpn, route-type
Ingress-Aliasing
                      to table mhevpn.evpn-mac.0
300144             *[EVPN/7] 02:27:06, routing-instance VS-1, route-type
Egress-MAC, ESI 00:44:44:44:44:44:44:44:44:44
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300160             *[EVPN/7] 02:29:22, routing-instance VS-1, route-type
Ingress-Aliasing
                      to table VS-1.evpn-mac.0
300176             *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
Egress-MAC, ESI 00:44:44:44:44:44:44:44:44:44
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300192             *[EVPN/7] 02:29:22, routing-instance VS-2, route-type
Ingress-Aliasing
                      to table VS-2.evpn-mac.0
300208             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-IM, vlan-id 120
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300224             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 mhevpn, route-type Egress-IM, vlan-id 10
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300240             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-IM, vlan-id 110
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
```

```
300256             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-IM, vlan-id 130
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300272             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-IM, vlan-id 210
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300288             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-IM, vlan-id 220
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300304             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-IM, vlan-id 230
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300320             *[EVPN/7] 02:27:06, routing-instance VS-1, route-type
Egress-MAC, ESI 00:11:11:11:11:11:11:11:11:11
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300336             *[EVPN/7] 02:27:06, routing-instance VS-1, route-type
Egress-MAC, ESI 00:33:33:33:33:33:33:33:33:33
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300368             *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
Egress-MAC, ESI 00:33:33:33:33:33:33:33:33:33
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300384             *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
Egress-MAC, ESI 00:11:11:11:11:11:11:11:11:11
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300416             *[EVPN/7] 02:27:06, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:33:33:33:33:33:33:33:33:33
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300432             *[EVPN/7] 02:27:06, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:11:11:11:11:11:11:11:11:11
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300480             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300496             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300560             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300592             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-MAC
```

```
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300608              *[EVPN/7] 02:29:23
                     > via ge-0/0/1.1001, Pop
300624              *[EVPN/7] 02:29:23
                     > via ge-0/0/1.2001, Pop
301232              *[EVPN/7] 02:29:17
                     > via ge-0/0/1.1002, Pop
301296              *[EVPN/7] 02:29:10
                     > via ge-0/0/1.1003, Pop
301312              *[EVPN/7] 02:27:06
                     > via ae10.2003, Pop
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301360              *[EVPN/7] 02:29:01
                     > via ge-0/0/1.1004, Pop
301408              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 vpws1004, route-type Egress, vlan-id 2004
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301456              *[EVPN/7] 02:27:06
                     > via ae10.1010, Pop
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301552              *[EVPN/7] 02:27:07, routing-instance VS-1, route-type
Egress-MAC, ESI 00:22:22:22:22:22:22:22:22:22
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301568              *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
Egress-MAC, ESI 00:22:22:22:22:22:22:22:22:22
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301648              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 vpws1010, route-type Egress, vlan-id 2010
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301664              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 mhevpn, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301680              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 mhevpn, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301696              *[EVPN/7] 02:27:07, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:22:22:22:22:22:22:22:22:22
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301712              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301728              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301744              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-IM, vlan-id 230
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301760              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 vpws1010, route-type Egress, vlan-id 2010
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301776              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301792              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-IM, vlan-id 130
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301808              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 vpws1004, route-type Egress, vlan-id 2004
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
```

```
301824             *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-IM, vlan-id 10
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301840             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1002, route-type Egress, vlan-id 2002
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301856             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1003, route-type Egress, vlan-id 2003
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301872             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1003, route-type Egress Protection, vlan-id 2003
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301888             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1010, route-type Egress Protection, vlan-id 1010
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301904             *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-IM, vlan-id 220
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301920             *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-IM, vlan-id 210
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301936             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-IM, vlan-id 230
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301952             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-SH, vlan-id 230
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301968             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-IM, vlan-id 220
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301984             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-SH, vlan-id 220
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302000             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-IM, vlan-id 210
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302016             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-SH, vlan-id 210
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302032             *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302048             *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302064             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302080             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302096             *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302112             *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302128             *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-MAC
```

```
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302144              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302160              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-IM, vlan-id 120
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302176              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-IM, vlan-id 110
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302192              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-IM, vlan-id 130
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302208              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-SH, vlan-id 130
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302224              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-IM, vlan-id 120
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302240              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-SH, vlan-id 120
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302256              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-IM, vlan-id 110
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302272              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-SH, vlan-id 110
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302288              *[EVPN/7] 02:27:06, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302304              *[EVPN/7] 02:27:06, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302320              *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302336              *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302352              *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 vpws1004, route-type Egress, vlan-id 2004
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302368              *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-IM, vlan-id 10
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302384              *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-SH, vlan-id 10
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302400              *[EVPN/7] 02:26:21
                      > via ge-0/0/1.3001, Pop
302432              *[EVPN/7] 02:26:21, remote-pe 100.100.100.3, routing-instance
 vpws3001, route-type Egress, vlan-id 40000
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302448              *[EVPN/7] 02:26:21, remote-pe 100.100.100.1, routing-instance
 vpws3001, route-type Egress, vlan-id 40000
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302464              *[EVPN/7] 02:26:20, remote-pe 100.100.100.2, routing-instance
 vpws3001, route-type Egress, vlan-id 40000
```

```
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302480             *[EVPN/7] 02:26:14
                    > via ge-0/0/1.3016, Pop
302512             *[EVPN/7] 02:26:14, remote-pe 100.100.100.1, routing-instance
 vpws3016, route-type Egress, vlan-id 40016
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302528             *[EVPN/7] 02:26:14, remote-pe 100.100.100.2, routing-instance
 vpws3016, route-type Egress, vlan-id 40016
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302560             *[EVPN/7] 02:26:06
                    > via ae10.3011, Pop
                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302592             *[EVPN/7] 02:26:07, remote-pe 100.100.100.1, routing-instance
 vpws3011, route-type Egress, vlan-id 401100
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302608             *[EVPN/7] 02:26:07, remote-pe 100.100.100.2, routing-instance
 vpws3011, route-type Egress, vlan-id 401100
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302624             *[EVPN/7] 02:26:07, remote-pe 100.100.100.3, routing-instance
 vpws3011, route-type Egress Protection, vlan-id 301100
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302656             *[EVPN/7] 02:25:59
                    > via ae10.3006, Pop
                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302688             *[EVPN/7] 02:26:00, remote-pe 100.100.100.2, routing-instance
 vpws3006, route-type Egress, vlan-id 400600
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302704             *[EVPN/7] 02:26:00, remote-pe 100.100.100.1, routing-instance
 vpws3006, route-type Egress, vlan-id 400600
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302720             *[EVPN/7] 02:25:59, remote-pe 100.100.100.3, routing-instance
 vpws3006, route-type Egress, vlan-id 400600
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302736             *[EVPN/7] 02:25:59, remote-pe 100.100.100.3, routing-instance
 vpws3006, route-type Egress Protection, vlan-id 300600
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ge-0/0/1.1001      *[EVPN/7] 02:29:23
                    > via ge-0/0/1.2001
ge-0/0/1.2001      *[EVPN/7] 02:29:23
                    > via ge-0/0/1.1001
ge-0/0/1.1002      *[EVPN/7] 02:27:06
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ae10.2003          *[EVPN/7] 02:29:10
                    > via ge-0/0/1.1003
ge-0/0/1.1003      *[EVPN/7] 02:27:06
                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3

                    > via ae10.2003
ge-0/0/1.1004      *[EVPN/7] 02:27:06
                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ae10.1010          *[EVPN/7] 02:27:06
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
ge-0/0/1.3001      *[EVPN/7] 02:26:20
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
```

```
                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ge-0/0/1.3016      *[EVPN/7] 02:26:13
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
ae10.3011          *[EVPN/7] 02:26:06
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
ae10.3006          *[EVPN/7] 02:25:59
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
```

show route table mpls.0 protocol ospf

user@host> **show route table mpls.0 protocol ospf**

```
mpls.0: 29 destinations, 29 routes (29 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

299952             *[L-OSPF/10] 23:59:42, metric 0
                    > to 10.0.10.70 via lt-1/2/0.14, Pop
                      to 10.0.6.60 via lt-1/2/0.12, Swap 800070, Push 800030(top)
299952(S=0)        *[L-OSPF/10] 23:59:42, metric 0
                    > to 10.0.10.70 via lt-1/2/0.14, Pop
                      to 10.0.6.60 via lt-1/2/0.12, Swap 800070, Push 800030(top)
299968             *[L-OSPF/10] 23:59:48, metric 0
                    > to 10.0.6.60 via lt-1/2/0.12, Pop
```

show route table mpls.0 extensive (PTX Series)

user@host> **show route table mpls.0 extensive**

```
ge-0/0/2.600 (1 entry, 1 announced)
TSI:
KRT in-kernel ge-0/0/2.600.0      /32 -> {composite(570)}
        *L2VPN  Preference: 7
                Next hop type: Indirect
                Address: 0x9438f34
                Next-hop reference count: 2
                Next hop type: Router, Next hop index: 567
                Next hop: 10.0.0.1 via ge-0/0/1.0, selected
                Label operation: Push 299808
                Label TTL action: prop-ttl
                Load balance label: Label 299808:None;
                Session Id: 0x1
                Protocol next hop: 10.255.255.1
                Label operation: Push 299872 Offset: 252
                Label TTL action: no-prop-ttl
                Load balance label: Label 299872:Flow label PUSH;
                Composite next hop: 0x9438ed8 570 INH Session ID: 0x2
                Indirect next hop: 0x9448208 262142 INH Session ID: 0x2
                State: <Active Int>
                Age: 47         Metric2: 1
                Validation State: unverified
```

```
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 2-Common L2 VC
                AS path: I
                Composite next hops: 1
                        Protocol next hop: 10.255.255.1 Metric: 1
                        Label operation: Push 299872 Offset: 252
                        Label TTL action: no-prop-ttl
                        Load balance label: Label 299872:Flow label PUSH;
                        Composite next hop: 0x9438ed8 570 INH Session ID: 0x2
                        Indirect next hop: 0x9448208 262142 INH Session ID: 0x2
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.0.0.1 via ge-0/0/1.0
                                Session Id: 0x1
                        10.255.255.1/32 Originating RIB: inet.3
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.0.0.1 via ge-0/0/1.0
```

show route table mpls.0 (RSVP Route—Transit LSP)

```
user@host> show route table mpls.0


mpls.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 00:37:31, metric 1
                      Receive
1                  *[MPLS/0] 00:37:31, metric 1
                      Receive
2                  *[MPLS/0] 00:37:31, metric 1
                      Receive
13                 *[MPLS/0] 00:37:31, metric 1
                      Receive
300352             *[RSVP/7/1] 00:08:00, metric 1
                   > to 10.64.0.106 via ge-1/0/1.0, label-switched-path lsp1_p2p
300352(S=0)        *[RSVP/7/1] 00:08:00, metric 1
                   > to 10.64.0.106 via ge-1/0/1.0, label-switched-path lsp1_p2p
300384             *[RSVP/7/2] 00:05:20, metric 1
                    > to 10.64.1.106 via ge-1/0/0.0, Pop
300384(S=0)        *[RSVP/7/2] 00:05:20, metric 1
                    > to 10.64.1.106 via ge-1/0/0.0, Pop
```

show route table vpls_1 detail

```
user@host> show route table vpls_1 detail

vpls_1.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

172.16.1.11:1000:1:1/96 (1 entry, 1 announced)
*L2VPN Preference: 170/-1
Receive table: vpls_1.l2vpn.0
Next-hop reference count: 2
State: <Active Int Ext>
```

```
Age: 4:29:47 Metric2: 1
Task: vpls_1-l2vpn
Announcement bits (1): 1-BGP.0.0.0.0+179
AS path: I
Communities: Layer2-info: encaps:VPLS, control flags:Site-Down
Label-base: 800000, range: 8, status-vector: 0xFF
```

show route table vpn-a

```
user@host> show route table vpn-a

vpn-a.l2vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

+ = Active Route, - = Last Active, * = Both
192.168.16.1:1:1:1/96
                   *[VPN/7] 05:48:27
                     Discard
192.168.24.1:1:2:1/96
                   *[BGP/170] 00:02:53, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
192.168.24.1:1:3:1/96
                   *[BGP/170] 00:02:53, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
```

show route table vpn-a.mdt.0

```
user@host> show route table vpn-a.mdt.0

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

1:1:0:10.255.14.216:232.1.1.1/144
                   *[MVPN/70] 01:23:05, metric2 1
                      Indirect
1:1:1:10.255.14.218:232.1.1.1/144
                   *[BGP/170] 00:57:49, localpref 100, from 10.255.14.218
                      AS path: I
                    > via so-0/0/0.0, label-switched-path r0e-to-r1
1:1:2:10.255.14.217:232.1.1.1/144
                   *[BGP/170] 00:57:49, localpref 100, from 10.255.14.217
                      AS path: I
> via so-0/0/1.0, label-switched-path r0-to-r2
```

show route table VPN-A detail

```
user@host>  show route table VPN-A detail

VPN-AB.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
10.255.179.9/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.179.13:200
                Next hop type: Indirect
                Next-hop reference count: 5
```

```
            Source: 10.255.179.13
            Next hop type: Router, Next hop index: 732
            Next hop: 10.39.1.14 via fe-0/3/0.0, selected
            Label operation: Push 299824, Push 299824(top)
            Protocol next hop: 10.255.179.13
            Push 299824
            Indirect next hop: 8f275a0 1048574
            State: (Secondary Active Int Ext)
            Local AS: 1 Peer AS: 1
            Age: 3:41:06 Metric: 1 Metric2: 1
            Task: BGP_1.10.255.179.13+64309
            Announcement bits (2): 0-KRT 1-BGP RT Background
            AS path: I
            Communities: target:1:200 rte-type:0.0.0.0:1:0
            Import Accepted
            VPN Label: 299824 TTL Action: vrf-ttl-propagate
            Localpref: 100
            Router ID: 10.255.179.13
            Primary Routing Table bgp.l3vpn.0
```

show route table VPN-AB.inet.0

```
user@host> show route table VPN-AB.inet.0
```

```
     VPN-AB.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
     + = Active Route, - = Last Active, * = Both

     10.39.1.0/30       *[OSPF/10] 00:07:24, metric 1
                         > via so-7/3/1.0
     10.39.1.4/30       *[Direct/0] 00:08:42
                         > via so-5/1/0.0
     10.39.1.6/32       *[Local/0] 00:08:46
                           Local
     10.255.71.16/32    *[Static/5] 00:07:24
                         > via so-2/0/0.0
     10.255.71.17/32    *[BGP/170] 00:07:24, MED 1, localpref 100, from
     10.255.71.15
                           AS path: I
                         > via so-2/1/0.0, Push 100020, Push 100011(top)
     10.255.71.18/32    *[BGP/170] 00:07:24, MED 1, localpref 100, from
     10.255.71.15
                           AS path: I
                         > via so-2/1/0.0, Push 100021, Push 100011(top)
     10.255.245.245/32  *[BGP/170] 00:08:35, localpref 100
                           AS path: 2 I
                         > to 10.39.1.5 via so-5/1/0.0
     10.255.245.246/32  *[OSPF/10] 00:07:24, metric 1
                         > via so-7/3/1.0
```

show route table VPN_blue.mvpn-inet6.0

```
user@host> show route table VPN_blue.mvpn-inet6.0
```

```
vpn_blue.mvpn-inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
1:10.255.2.202:65536:10.255.2.202/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.202
                      AS path: I
                    > via so-0/1/3.0
1:10.255.2.203:65536:10.255.2.203/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.203
                      AS path: I
                    > via so-0/1/0.0
1:10.255.2.204:65536:10.255.2.204/432
                   *[MVPN/70] 00:57:23, metric2 1
                      Indirect
5:10.255.2.202:65536:128:::192.168.90.2:128:ffff::1/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.202
                      AS path: I
                    > via so-0/1/3.0
6:10.255.2.203:65536:64500:128:::10.12.53.12:128:ffff::1/432
                   *[PIM/105] 00:02:37
                      Multicast (IPv6)
7:10.255.2.202:65536:64500:128:::192.168.90.2:128:ffff::1/432
                   *[MVPN/70] 00:02:37, metric2 1
                      Indirect
```

show route table vrf1.mvpn.0 extensive

```
user@host> show route table vrf1.mvpn.0 extensive
```

```
1:10.255.50.77:1:10.255.50.77/240 (1 entry, 1 announced)
        *MVPN   Preference: 70
                PMSI: Flags 0x0: Label 0: RSVP-TE:
Session_13[10.255.50.77:0:25624:10.255.50.77]
                Next hop type: Indirect
                Address: 0xbb2c944
                Next-hop reference count: 360
                Protocol next hop: 10.255.50.77
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 53:03      Metric2: 1
                Validation State: unverified
                Task: mvpn global task
                Announcement bits (3): 0-PIM.vrf1 1-mvpn global task 2-rt-export

                AS path: I
```

show route table inetflow detail

```
user@host> show route table inetflow detail
```

```
inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.12.44.1,*/48 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next-hop reference count: 2
                State: <Active Ext>
                Local AS: 64502 Peer AS: 64500
                Age: 4
                Task: BGP_64500.10.12.99.5+3792
                Announcement bits (1): 0-Flow
```

```
                AS path: 64500 I
                Communities: traffic-rate:0:0
                Validation state: Accept, Originator: 10.12.99.5
                Via: 10.12.44.0/24, Active
                Localpref: 100
                Router ID: 10.255.71.161

10.12.56.1,*/48 (1 entry, 1 announced)
        *Flow   Preference: 5
                Next-hop reference count: 2
                State: <Active>
                Local AS: 64502
                Age: 6:30
                Task: RT Flow
                Announcement bits (2): 0-Flow 1-BGP.0.0.0.0+179
                AS path: I
                Communities: 1:1
```

user@host> **show route table green.l2vpn.0** (VPLS Multihoming with FEC 129)

```
green.l2vpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.2:100:10.1.1.2/96 AD
                   *[VPLS/170] 1d 03:11:03, metric2 1
                         Indirect
10.1.1.4:100:10.1.1.4/96 AD
                   *[BGP/170] 1d 03:11:02, localpref 100, from 10.1.1.4
                      AS path: I, validation-state: unverified
                    >    via ge-1/2/1.5
10.1.1.2:100:1:0/96 MH
                   *[VPLS/170] 1d 03:11:03, metric2 1
                         Indirect
10.1.1.4:100:1:0/96 MH
                   *[BGP/170] 1d 03:11:02, localpref 100, from 10.1.1.4
                      AS path: I, validation-state: unverified
                    >    via ge-1/2/1.5
10.1.1.4:NoCtrlWord:5:100:100:10.1.1.2:10.1.1.4/176
                   *[VPLS/7] 1d 03:11:02, metric2 1
                    >    via ge-1/2/1.5
10.1.1.4:NoCtrlWord:5:100:100:10.1.1.4:10.1.1.2/176
                   *[LDP/9] 1d 03:11:02
                         Discard
```

user@host> **show route table red extensive**

```
red.inet.0: 364481 destinations, 714087 routes (364480 active, 48448 holddown, 1
 hidden)
10.0.0.0/32 (3 entries, 1 announced)
        State: <OnList CalcForwarding>
TSI:
KRT in-kernel 10.0.0.0/32 -> {composite(1048575)} Page 0 idx 1 Type 1 val 0x934342c

    Nexthop: Self
    AS path: [2] I
    Communities: target:2:1
```

```
Path 10.0.0.0 from 10.3.0.0 Vector len 4.  Val: 1
        @BGP    Preference: 170/-1
                Route Distinguisher: 2:1
                Next hop type: Indirect
                Address: 0x258059e4
                Next-hop reference count: 2
                Source: 2.2.0.0
                Next hop type: Router
                Next hop: 10.1.1.1 via ge-1/1/9.0, selected
                Label operation: Push 707633
                Label TTL action: prop-ttl
                Session Id: 0x17d8
                Protocol next hop: 10.2.0.0
                Push 16
                Composite next hop: 0x25805988 - INH Session ID: 0x193c
                Indirect next hop: 0x23eea900 - INH Session ID: 0x193c
                State: <Secondary Active Int Ext ProtectionPath ProtectionCand>
                Local AS:     2 Peer AS:     2
                Age: 23         Metric2: 35
                Validation State: unverified
                Task: BGP_172.16.2.0.0+34549
                AS path: I
                Communities: target:2:1
                Import Accepted
                VPN Label: 16
                Localpref: 0
                Router ID: 10.2.0.0
                Primary Routing Table bgp.l3vpn.0
                Composite next hops: 1
                        Protocol next hop: 10.2.0.0 Metric: 35
                        Push 16
                        Composite next hop: 0x25805988 - INH Session ID: 0x193c
                        Indirect next hop: 0x23eea900 - INH Session ID: 0x193c
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.1.1.1 via ge-1/1/9.0
                                Session Id: 0x17d8
                        2.2.0.0/32 Originating RIB: inet.3
                          Metric: 35                    Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.1.1.1 via ge-1/1/9.0
         BGP    Preference: 170/-1
                Route Distinguisher: 2:1
                Next hop type: Indirect
                Address: 0x9347028
                Next-hop reference count: 3
                Source: 10.3.0.0
                Next hop type: Router, Next hop index: 702
                Next hop: 10.1.4.2 via ge-1/0/0.0, selected
                Label operation: Push 634278
                Label TTL action: prop-ttl
                Session Id: 0x17d9
                Protocol next hop: 10.3.0.0
                Push 16
                Composite next hop: 0x93463a0 1048575 INH Session ID: 0x17da
                Indirect next hop: 0x91e8800 1048574 INH Session ID: 0x17da
                State: <Secondary NotBest Int Ext ProtectionPath ProtectionCand>

                Inactive reason: Not Best in its group - IGP metric
```

```
               Local AS:     2 Peer AS:     2
               Age: 3:34       Metric2: 70
               Validation State: unverified
               Task: BGP_172.16.3.0.0+32805
               Announcement bits (2): 0-KRT 1-BGP_RT_Background
               AS path: I
               Communities: target:2:1
               Import Accepted
               VPN Label: 16
               Localpref: 0
               Router ID: 10.3.0.0
               Primary Routing Table bgp.l3vpn.0
               Composite next hops: 1
                       Protocol next hop: 10.3.0.0 Metric: 70
                       Push 16
                       Composite next hop: 0x93463a0 1048575 INH Session ID:
0x17da
                       Indirect next hop: 0x91e8800 1048574 INH Session ID:
0x17da
                       Indirect path forwarding next hops: 1
                               Next hop type: Router
                               Next hop: 10.1.4.2 via ge-1/0/0.0
                               Session Id: 0x17d9
                       10.3.0.0/32 Originating RIB: inet.3
                         Metric: 70                     Node path count: 1
                         Forwarding nexthops: 1
                               Nexthop: 10.1.4.2 via ge-1/0/0.0
       #Multipath Preference: 255
               Next hop type: Indirect
               Address: 0x24afca30
               Next-hop reference count: 1
               Next hop type: Router
               Next hop: 10.1.1.1 via ge-1/1/9.0, selected
               Label operation: Push 707633
               Label TTL action: prop-ttl
               Session Id: 0x17d8
               Next hop type: Router, Next hop index: 702
               Next hop: 10.1.4.2 via ge-1/0/0.0
               Label operation: Push 634278
               Label TTL action: prop-ttl
               Session Id: 0x17d9
               Protocol next hop: 10.2.0.0
               Push 16
               Composite next hop: 0x25805988 - INH Session ID: 0x193c
              Indirect next hop: 0x23eea900 - INH Session ID: 0x193c Weight 0x1

               Protocol next hop: 10.3.0.0
               Push 16
               Composite next hop: 0x93463a0 1048575 INH Session ID: 0x17da
             Indirect next hop: 0x91e8800 1048574 INH Session ID: 0x17da Weight
 0x4000
               State: <ForwardingOnly Int Ext>
               Inactive reason: Forwarding use only
               Age: 23         Metric2: 35
               Validation State: unverified
               Task: RT
               AS path: I
               Communities: target:2:1
```

show route table bgp.evpn.0 extensive |no-more (EVPN)

```
show route table bgp.evpn.0 extensive | no-more

bgp.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
2:1000:10::100::00:aa:aa:aa:aa:aa/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
         State: Local AS: 17  Peer AS:17  Age:21:12  Metric2:1  Validation State:
 unverified
                Task: BGP_17.1.2.3.4+50756
                AS path: I
                Communities: target:1111:8388708 encapsulation0:0:0:0:3
                Import Accepted
                Route Label: 100
                ESI: 00:00:00:00:00:00:00:00:00:00
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.10.10.1 via xe-0/0/1.0
                                Session Id: 0x2
                        1.2.3.4/32 Originating RIB: inet.0
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.92.78.102 via em0.0

2:1000:10::200::00:bb:bb:bb:bb:bb/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
          State: Local AS:17  Peer AS:17  Age:19:43  Metric2:1  Validation
State:unverified
                Task: BGP_17.1.2.3.4+50756
                AS path: I
                Communities: target:2222:22 encapsulation0:0:0:0:3
                Import Accepted
                Route Label: 200
                ESI: 00:00:00:00:00:00:00:00:00:00
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
```

```
                      Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                      Indirect path forwarding next hops: 1
                              Next hop type: Router
                              Next hop: 10.10.10.1 via xe-0/0/1.0
                              Session Id: 0x2
                      10.2.3.4/32 Originating RIB: inet.0
                        Metric: 1                       Node path count: 1
                        Forwarding nexthops: 2
                              Nexthop: 10.92.78.102 via em0.0

2:1000:10::300::00:cc:cc:cc:cc:cc/304 (1 entry, 0 announced)
       *BGP    Preference: 170/-101
               Route Distinguisher: 1000:10
               Next hop type: Indirect
               Address: 0x9420fd0
               Next-hop reference count: 12
               Source: 10.2.3.4
               Protocol next hop: 10.2.3.4
               Indirect next hop: 0x2 no-forward INH Session ID: 0x0
        State: Local AS:17  Peer AS:17  Age:17:21  Metric2:1  Validation State:
 unverified  Task: BGP 17,1,2,3,3,4+50756
         AS path: I
               Communities: target:3333:33 encapsulation0:0:0:0:3
               Import Accepted
               Route Label: 300
               ESI: 00:00:00:00:00:00:00:00:00:00
               Localpref: 100
               Router ID: 10.2.3.4
               Secondary Tables: default-switch.evpn.0
               Indirect next hops: 1
                       Protocol next hop: 10.2.3.4 Metric: 1
                       Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                       Indirect path forwarding next hops: 1
                               Next hop type: Router
                               Next hop: 10.10.10.1 via xe-0/0/1.0
                               Session Id: 0x2
                       10.2.3.4/32 Originating RIB: inet.0
                         Metric: 1                       Node path count: 1
                         Forwarding nexthops: 2
                               Nexthop: 10.92.78.102 via em0.0

3:1000:10::100::1.2.3.4/304 (1 entry, 0 announced)
       *BGP    Preference: 170/-101
               Route Distinguisher: 1000:10
               PMSI: Flags 0x0: Label 100: Type INGRESS-REPLICATION 1.2.3.4
               Next hop type: Indirect
               Address: 0x9420fd0
               Next-hop reference count: 12
               Source: 10.2.3.4
               Protocol next hop: 10.2.3.4
               Indirect next hop: 0x2 no-forward INH Session ID: 0x0
        State: Local AS:17  Peer AS:17 Age:37:01  Metric2:1  Validation State:
 unverified  Task: BGP 17.1.2.3.4+50756
               AS path: I
               Communities: target:1111:8388708 encapsulation0:0:0:0:3
               Import Accepted
               Localpref: 100
               Router ID: 10.2.3.4
               Secondary Tables: default-switch.evpn.0
```

```
               Indirect next hops: 1
                       Protocol next hop: 10.2.3.4 Metric: 1
                       Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                       Indirect path forwarding next hops: 1
                               Next hop type: Router
                               Next hop: 10.10.10.1 via xe-0/0/1.0
                               Session Id: 0x2
                       10.2.3.4/32 Originating RIB: inet.0
                         Metric: 1                       Node path count: 1
                         Forwarding nexthops: 2
                               Nexthop: 10.92.78.102 via em0.0

3:1000:10::200::1.2.3.4/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                PMSI: Flags 0x0: Label 200: Type INGRESS-REPLICATION 1.2.3.4
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
         State: Local AS: 17 Peer AS: 17   Age:35:22  Metric2:1  Validation
State:unverified  Task: BGP 17.1.2.3.4+50756
          AS path:I  Communities: target:2222:22 encapsulation):0:0:0:0:3

Import Accepted
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                       Protocol next hop: 10.2.3.4 Metric: 1
                       Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                       Indirect path forwarding next hops: 1
                               Next hop type: Router
                               Next hop: 10.10.10.1 via xe-0/0/1.0
                               Session Id: 0x2
                       10.2.3.4/32 Originating RIB: inet.0
                         Metric: 1                       Node path count: 1
                         Forwarding nexthops: 2
                               Nexthop: 10.92.78.102 via em0.0

3:1000:10::300::1.2.3.4/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                PMSI: Flags 0x0: Label 300: Type INGRESS-REPLICATION 1.2.3.4
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
         State: Local AS: 17 Peer AS:  17  Age 35:22 Metric2:1  Validation State:
 unverified   Task: BGP 17.1.2.3.4+5075
          6  AS path: I   Communities: target:3333:33  encapsulation0:0:0:0:3
Import Accepted   Localpref:100
          Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
```

```
                    Protocol next hop: 10.2.3.4 Metric: 1
                    Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                    Indirect path forwarding next hops: 1
                            Next hop type: Router
                            Next hop: 10.10.10.1 via xe-0/0/1.0
                            Session Id: 0x2
                    10.2.3.4/32 Originating RIB: inet.0
                      Metric: 1                       Node path count: 1
                      Forwarding nexthops: 2
                            Nexthop: 10.92.78.102 via em0.0
```

show vlans evpn nd-table

Syntax

```
show vlans evpn nd-table
<brief | detail | extensive>
<count>
<instance-name>
<mac-address>
<vlan-name>
```

Release Information Command introduced in Junos OS Release 17.3R1 for EX Series Switches.

NOTE: Starting with Junos OS Release 17.4R2, the **show vlans evpn nd-table** command is replaced by the show ethernet-switching evpn nd-table command.

Description Display information about INET entries associated with MAC addresses learned through Network Discovery Protocol (NDP).

Options

none—Display information for all INET entries.

brief | detail | extensive—(Optional) Display the specified level of output.

count—(Optional) Display the number of INET addresses learned in a routing instance.

instance—(Optional) Display information for a specified instance.

mac-adress—(Optional) Display information for a specified MAC address.

vlan-name (all)—(Optional) Display information for a specified VLAN or for all VLANs.

Required Privilege Level view

List of Sample Output

show vlans evpn nd-table brief on page 1259
show vlans evpn nd-table detail on page 1259
show vlans evpn nd-table extensive on page 1259
show vlans evpn nd-table count on page 1260
show vlans evpn nd-table instance evpn1 on page 1260
show vlans evpn nd-table instance 00:05:86:90:bd:f0 (MAC address) on page 1260
show vlans evpn nd-table vlan-name vlan10 on page 1260

Output Fields Table 28 on page 1122 lists the output fields for the **show vlans evpn nd-table** command. Output fields are listed in the approximate order in which they appear.

Table 47: show vlans evpn nd-table Output Fields

Field Name	Field Description	Level of Output
INET address	The INET address related to the INET entries that are added to the NDP table.	All levels
MAC address	MAC addresses learned through NDP.	**brief**, **detail**, **extensive**, **instance**, **mac-addressvlan-name**,,
Logical Interface	Logical interface associated with the routing instance in which the NDP INET address is learned.	**brief**, **instance**, **mac-addressvlan-name**,,
Routing instance	Routing instance in which the NDP INET address is learned.	all levels
Bridging domain	Bridging domain in which the NDP INET address is learned.	all levels
Learning interface	Interface on which the NDP INET address is learned.	**detail**, **extensive**
Count	Indicates the number of NDP INET addresses learned in a routing instance in a bridge domain.	**count**

Sample Output

show vlans evpn nd-table brief

```
user@switch> show vlans evpn nd-table brief

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2            00:05:86:a0:d5:00 irb.0         evpn1      vlan10
```

show vlans evpn nd-table detail

```
user@switch> show vlans evpn nd-table detail

INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
   Bridging domain: vlan10
   Learning interface: irb.0
```

show vlans evpn nd-table extensive

```
user@switch> show vlans evpn nd-table extensive

INET address: 8002::2
MAC address: 00:05:86:a0:d5:00
  Routing instance: evpn1
   Bridging domain: vlan10
   Learning interface: irb.0
```

show vlans evpn nd-table count

```
user@switch> show vlans evpn nd-table count

1 ND INET addresses learned in routing instance evpn1 bridge domain vlan10
```

show vlans evpn nd-table instance evpn1

```
user@switch> show vlans evpn nd-table instance evpn1

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2
           00:05:86:a0:d5:00
  irb.0         evpn1        vlan10
```

show vlans evpn nd-table instance 00:05:86:90:bd:f0 (MAC address)

```
user@switch> show vlans evpn nd-table

INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2
           00:05:86:a0:d5:00
  irb.0         evpn1        __evpn1__
```

show vlans evpn nd-table vlan-name vlan10

```
user@switch>show vlans evpn nd-table vlan-name vlan10


INET               MAC               Logical       Routing      Bridging
address            address           interface     instance     domain
8002::2           00:05:86:a0:d5:00
  irb.0         evpn1        vlan10
```