JUNIPER NETWORKS

Junos® OS

OVSDB and VXLAN Feature Guide

Release
14.2

Published: 2014-10-21

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Junos® OS OVSDB and VXLAN Feature Guide
14.2

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

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 http://www.juniper.net/techpubs/.

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 http://www.juniper.net/books.

Supported Platforms

For the features described in this document, the following platforms are supported:

- EX Series
- MX Series
- QFX Series standalone switches

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 the *CLI User Guide*.

Documentation Conventions

Table 1 on page xiii 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 xiv 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`
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

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
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, comments, and suggestions so that we can improve the documentation. You can provide feedback by using either of the following methods:

- Online feedback rating system—On any page at the Juniper Networks Technical Documentation site at http://www.juniper.net/techpubs/index.html, simply click the stars to rate the content, and use the pop-up form to provide us with information about your experience. Alternately, you can use the online feedback form at https://www.juniper.net/cgi-bin/docbugreport/.
- 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 JNASC 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 http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit http://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: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: http://kb.juniper.net/InfoCenter/
- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

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

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 http://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 http://www.juniper.net/support/requesting-support.html.

PART 1

Overview

CHAPTER 1

OVSDB Overview

Open vSwitch Database Support on Juniper Networks Devices

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Table 3 on page 3 lists the Juniper Networks devices that support the Open vSwitch Database (OVSDB) management protocol. For each device, the table also includes the OVSDB software package and the initial Junos OS release that must be installed for OVSDB support. The OVSDB software package release must be the same as the Junos OS release running on the device.

Table 3: OVSDB Support on Junos OS Devices

Junos OS Device	OVSDB Software Package	Junos OS Release
MX80 3D Universal Edge Router	jsdn-powerpc-*release*	14.1R2
MX240, MX480, MX960 3D Universal Edge Routers	jsdn-i386-*release*	14.1R2
QFX5100 Ethernet Switch	jsdn-i386-*release*	14.1X53-D10
EX9200 Ethernet Switch	jsdn-i386-*release*	14.2

Related Documentation

- Installing Open vSwitch Database Components on Juniper Networks Devices on page 79

Understanding the Junos OS Implementation of VXLAN and OVSDB in a VMware NSX for Multi-Hypervisor Environment for the Data Center

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Some Juniper Networks devices support Virtual Extensible LAN (VXLAN) and the Open vSwitch Database (OVSDB) management protocol. (For information about the Juniper Networks devices on which OVSDB is supported and the Juniper Networks Junos operating system (Junos OS) release in which support is introduced, see "Open vSwitch Database Support on Juniper Networks Devices" on page 3.) Support for VXLAN and OVSDB enables the Juniper Networks devices in a physical network to be integrated into a virtual network.

The implementation of VXLAN and OVSDB on Juniper Networks devices is supported in a VMware NSX for Multi-Hypervisor environment for the data center. Table 4 on page 4 outlines the components that compose this environment and products that are typically deployed for each component.

Table 4: NSX for Multi-Hypervisor Components and Products That Can Be Implemented

Component	Products
Cloud management platform (CMP)	CloudStack OpenStack Custom CMP
Network virtualization platform	NSX for Multi-Hypervisor
Hypervisor	Kernel-based Virtual Machine (KVM) Red Hat VMware ESXi Xen NOTE: Juniper Networks supports only KVM and ESXi.
Virtual switch	Open vSwitch (OVS) NSX vSwitch
SDN controller	NSX for Multi-Hypervisor controller
Overlay protocol	VXLAN
MAC learning protocol	OVSDB

Figure 1 on page 5 shows a high-level view of the architecture into which the NSX for Multi-Hypervisor platform fits, while Figure 2 on page 5 provides a more detailed representation of the components in the virtual and physical networks.

Figure 1: High-Level NSX for Multi-Hypervisor Architecture

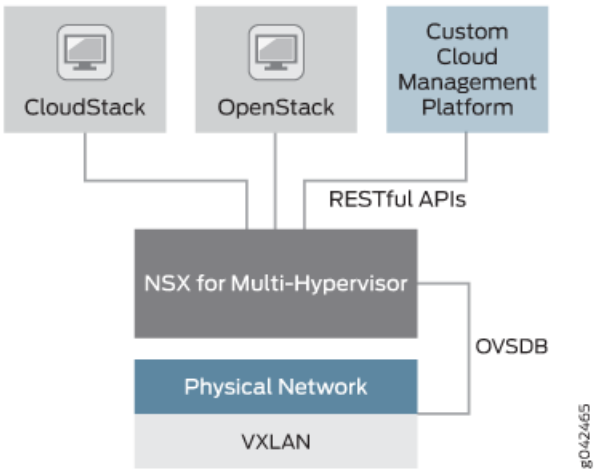
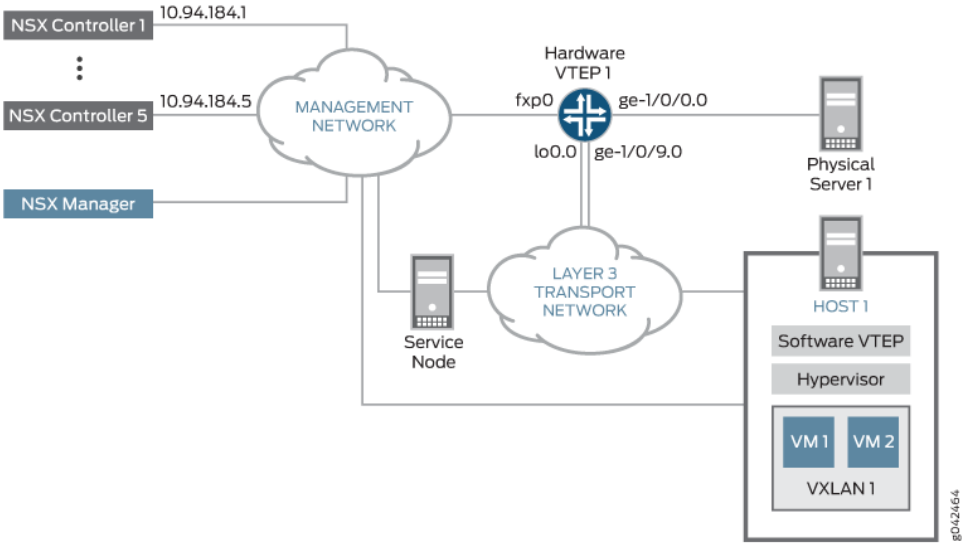


Figure 2: Integration of Juniper Networks Device That Implements VXLAN and OVSDB into NSX for Multi-Hypervisor Environment



In the data center topology shown in Figure 2 on page 5, the physical and virtual servers need to communicate. To facilitate this communication, a Juniper Networks device that supports VXLAN is strategically deployed so that it serves as a *gateway*, which is also known as a hardware virtual tunnel endpoint (VTEP), at the edge of the physical network. Working in conjunction with the software VTEP, which is deployed at the edge of the virtual network, the hardware VTEP encapsulates packets from resources on Physical

Server 1 with a VXLAN header, and after the packets traverse the Layer 3 transport network, the software VTEP removes the VXLAN header from the packets and forwards the packets to the appropriate virtual machines (VMs). In essence, the encapsulation and de-encapsulation of packets by the hardware and software VTEPs enables components in the physical and virtual networks to coexist without one needing to understand the workings of the other.

The same Juniper Networks device that acts as hardware VTEP in Figure 2 on page 5 implements OVSDB, which enables this device to learn the MAC addresses of Physical Server 1 and other physical servers, and publish the addresses in the OVSDB schema, which was defined for physical devices. In the virtual network, one or more NSX controllers collect the MAC addresses of HOST 1 and other virtual servers, and publish the addresses in the OVSDB schema. Using the OVSDB schema, components in the physical and virtual networks can exchange MAC addresses, as well as statistical information, enabling the components to learn about and reach each other in their respective networks.

Related Documentation

- Understanding the Open vSwitch Database Management Protocol Running on Juniper Networks Devices on page 6
- Open vSwitch Database Schema For Physical Devices on page 12

Understanding the Open vSwitch Database Management Protocol Running on Juniper Networks Devices

Supported Platforms EX Series, MX Series, QFX Series standalone switches

The Juniper Networks Junos operating system (Junos OS) implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which VMware NSX controllers and Juniper Networks devices that support OVSDB can communicate. In an NSX multi-hypervisor environment, NSX controllers and Juniper Networks devices exchange control and statistical information, thereby enabling virtual machine (VM) traffic from entities in a virtual network to be forwarded to entities in a physical network and vice versa.

To enable communication between NSX controllers and Juniper Networks devices, the Junos OS implementation of OVSDB includes an OVSDB server and an OVSDB client, both of which run on each Juniper Networks device that supports OVSDB.

The OVSDB server on a Juniper Networks device can communicate with an OVSDB client on one or more NSX controllers. To establish a connection between a Juniper Networks device and an NSX controller, you must specify information about the controller (IP address) and the connection (port over which the connection occurs and the communication protocol to be used) on each Juniper Networks device. After the configuration is successfully committed, the connection is established between the management port of the Juniper Networks device and the NSX controller port that you specify in the Junos OS configuration.

The OVSDB server stores and maintains an OVSDB database schema, which is defined for physical devices. This schema contains control and statistical information provided by the OVSDB client on the Juniper Networks devices and NSX controllers. This information

is stored in various tables in the schema. The OVSDB client on the Juniper Networks devices and NSX controllers monitors the schema for additions, deletions, and modifications to this information, and the information is used for various purposes such as learning the MAC addresses of virtual hosts and physical servers.

The schema provides a means through which the Juniper Networks devices and the NSX controllers can exchange information. For example, the Juniper Networks devices capture MAC routes to entities in the physical network and push this information to a table in the schema so that NSX controllers with connections to these Juniper Networks devices can access the MAC routes. Conversely, NSX controllers capture MAC routes to entities in the virtual network and push this information to a table in the schema so that Juniper Networks devices with connections to the NSX controllers can access the MAC routes.

Some of the OVSDB table names include the words *local* or *remote*, for example, *unicast MACs local table* and *unicast MACs remote table*. Information in *local* tables is learned by a Juniper Networks device that functions as a hardware virtual tunnel endpoint (VTEP), while information in *remote* tables is learned from other software or hardware VTEPs.

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7
- Open vSwitch Database Schema For Physical Devices on page 12

Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers

Supported Platforms EX Series, MX Series, QFX Series standalone switches

The Juniper Networks Junos operating system (Junos OS) implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which VMware NSX controllers and Juniper Networks devices that support OVSDB can communicate. This implementation of OVSDB supports one cluster of NSX controllers, which includes three or five controllers as recommended by VMware.

To implement the OVSDB management protocol on a Juniper Networks device, you must explicitly configure a connection to one NSX controller, using the Junos OS CLI. If the NSX controller to which you explicitly configure a connection is in a cluster, the controller pushes information about other controllers in the same cluster to the device, and the device establishes connections with the other controllers. However, you can also explicitly configure connections with the other controllers in the cluster, using the Junos OS CLI.

A Juniper Networks device exchanges control and statistical data with each NSX controller to which it is connected. Therefore, the benefits of connecting a Juniper Networks device to multiple NSX controllers include redundancy and load-balancing of the controller workload.

Connections to all NSX controllers are made on the management interface of the Juniper Networks device. (The management interface on MX Series routers is fxp0 and on EX9200 switches is me0.) To set up a connection between a Juniper Networks device and an NSX controller, you need to configure the following parameters on the Juniper Networks device:

- IP address of the NSX controller.
- The protocol that secures the connection. Secure Sockets Layer (SSL) is the supported protocol.

NOTE: The SSL connection requires a private key and certificates, which must be stored in the /var/db/certs directory of the Juniper Networks device. For more information about the files, including actions you must take to create and install some of the files, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers" on page 80.

- Number of the port over which the connection is made. The port number of the default port is 6632.

Optionally, you can configure the following connection timers on the Juniper Networks device:

- Inactivity probe duration—The maximum amount of time, in milliseconds, that the connection can be inactive before an inactivity probe is sent. The default value is 0 milliseconds, which means that an inactivity probe is never sent.
- Maximum backoff duration—If an attempt to connect to an NSX controller fails, the maximum amount of time, in milliseconds, before the device can make the next attempt. The default value is 1000 milliseconds.

Related Documentation

- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81
- Understanding the Open vSwitch Database Management Protocol Running on Juniper Networks Devices on page 6

Understanding How Layer 2 BUM Traffic and Layer 3 Routed Multicast Traffic Are Handled in VXLANs Managed by OVSDB

Supported Platforms EX Series, MX Series

The Juniper Networks Junos operating system (Junos OS) implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which VMware NSX controllers and Juniper Networks devices that support OVSDB can communicate.

This topic explains how a Juniper Networks device with Virtual Extensible LAN (VXLAN) and OVSDB management protocol capabilities handles the following types of traffic:

- Layer 2 broadcast, unknown unicast, and multicast (BUM) traffic that originates in an OVSDB-managed VXLAN and is forwarded to interfaces within the same VXLAN.
- Layer 3 multicast traffic that is received by an integrated routing and bridging (IRB) interface in an OVSDB-managed VXLAN and is forwarded to interfaces in another OVSDB-managed VXLAN.

By default, Layer 2 BUM traffic that originates in an OVSDB-managed VXLAN is handled by one or more service nodes in the same VXLAN. When this option is used, the table for remote multicast MAC addresses in the OVSDB schema for physical devices contains only one entry that has the keyword **unknown-dst** as the MAC string and a list of software virtual tunnel endpoints (VTEPs) that host the service nodes.

Given the previously described table entry, Layer 2 BUM traffic received on an interface in the OVSDB-managed VXLAN is forwarded to one of the software VTEPs. The software VTEP, and therefore, the service node to which a BUM packet is forwarded, is determined by the Juniper Networks device on which the OVSDB-managed VXLAN is configured. On receiving the BUM packet, the service node replicates the packet and forwards the replicas to all interfaces within the VXLAN.

Instead of using service nodes, you can optionally enable ingress node replication to handle Layer 2 BUM traffic on Juniper Networks devices that support OVSDB.

With ingress node replication enabled, on receiving a Layer 2 BUM packet on an interface in an OVSDB-managed VXLAN, the Juniper Networks device replicates the packet and then forwards the replicas to all software VTEPs included in the unicast MACs remote table in the OVSDB schema. The software VTEPs then forward the replicas to all virtual machines (VMs), except service VMs or nodes, on the same host.

NOTE: When Juniper Networks devices replicate Layer 2 BUM packets to a large number of remote software VTEPs, the performance of the Juniper Networks devices can be impacted.

On IRB interfaces that forward Layer 3 multicast traffic from one OVSDB-managed VXLAN to another, ingress node replication is automatically implemented. With ingress node replication, the Juniper Networks device replicates a Layer 3 multicast packet and then the IRB interface forwards the replicas to all hardware and software VTEPs, but not to service nodes, in the other OVSDB-managed VXLAN. For the routing of Layer 3 multicast traffic from one OVSDB-managed VXLAN to another, ingress node replication is the only option and does not need to be configured.

Related Documentation

- Configuring OVSDB-Managed VXLANs on page 84
- Example: Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center on page 32
- Open vSwitch Database Schema For Physical Devices on page 12

Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment

Supported Platforms EX Series, MX Series, QFX Series

The Juniper Networks Junos operating system (Junos OS) implementation of the Open vSwitch Database (OVSDB) management protocol provides a means through which VMware NSX controllers and Juniper Networks devices that support OVSDB can communicate.

In a Junos OS environment, the concept of an OVSDB-managed Layer 2 broadcast domain in which data flows are limited to that domain is known as a *VXLAN*. In an NSX environment, the same concept is known as a *logical switch*. Understanding the different terminology in turn enables you to better understand the configuration tasks required for setting up OVSDB-managed VXLANs.

The following sections explain what you need to do to set up OVSDB-managed VXLANs properly for each Juniper Networks device that supports OVSDB and VXLAN:

- Understanding How to Set Up OVSDB-Managed VXLANs On Juniper Networks Devices on page 10
- Understanding How to Determine the State of an OVSDB-Managed VXLAN on page 11

Understanding How to Set Up OVSDB-Managed VXLANs On Juniper Networks Devices

For each VXLAN that you plan to implement, you must first configure a logical switch, using NSX Manager or the NSX API. Based on the name and the VXLAN network identifier (VNI) that you specify, NSX automatically generates a universally unique identifier (UUID) for the logical switch. You must retain the UUID of the logical switch for later use.

Next, on the Juniper Networks device, you must configure the corresponding VXLAN, including the same VNI specified for the logical switch, using the Junos OS CLI. For the name of the VXLAN, you must specify the UUID for the logical switch.

When configuring a logical switch and a corresponding VXLAN, it is important that the UUID and VNI in both configurations are the same. If these elements are not the same, the logical switch and VXLAN cannot become operational, which means they cannot exchange MAC addresses learned in the NSX and Junos OS environments, respectively.

Table 5 on page 11 provides a summary of the procedure that you must perform for each OVSDB-managed VXLAN on each Juniper Networks device, where to get more information about the configuration task, and the configuration statements that you must use to configure the VXLAN.

Table 5: Summary of Configuration Tasks for Setting Up An OVSDB-Managed VXLAN on Juniper Networks Devices

Juniper Networks Device That Supports OVSDB and VXLAN	Configure Logical Switch, Using NSX Manager or the NSX API?	Where to Find More Configuration Information	Configure Corresponding VXLAN on Juniper Networks Device?	Junos OS Statement to Configure the OVSDB-Managed VXLAN	Where to Find More Configuration Information
MX Series routers	Yes	See the documentation that accompanies NSX Manager or the NSX API.	Yes	**ovsdb-managed** statement in the **[edit bridge-domains *domain-name* vxlan]** hierarchy. For the name of the VXLAN, specify the UUID for the logical switch configured in NSX Manager or in the NSX API.	"Configuring OVSDB-Managed VXLANs" on page 84
EX9200 switch	Yes	See the documentation that accompanies NSX Manager or the NSX API.	Yes	**ovsdb-managed** statement in the **[edit vlans *vlan-name* vxlan]** hierarchy. For the name of the VXLAN, specify the UUID for the logical switch configured in NSX Manager or in the NSX API.	"Configuring OVSDB-Managed VXLANs" on page 84

Understanding How to Determine the State of an OVSDB-Managed VXLAN

Regardless of the Juniper Networks device and the procedure that you follow to set up OVSDB-managed VXLANs, after configuring one or more logical switches in NSX Manager or in the NSX API, the NSX controller pushes relevant information to the logical switch table in the OVSDB schema for physical devices, which resides on the respective devices.

To determine the state of a VXLAN and corresponding logical switch, you can use the **show ovsdb logical-switch** command. The following are possible states:

Created by Controller—A logical switch was configured in NSX Manager or in the NSX API, but the corresponding VXLAN is not yet created on the Juniper Networks device. In this state, the VXLAN and corresponding logical switch are not yet operational.

Created by L2ALD—A VXLAN was created, but the corresponding logical switch is not yet configured in NSX Manager or in the NSX API. In this state, the VXLAN and corresponding logical switch are not yet operational.

Created by both—A logical switch was configured in NSX Manager or in the NSX API, and a corresponding VXLAN was created. In this state, the VXLAN and corresponding logical switch are operational.

Tunnel key mismatch—The VNIs specified in the logical switch and corresponding VXLAN configurations do not match. In this state, the VXLAN and corresponding logical switch are not yet operational.

Related Documentation

- show ovsdb logical-switch on page 117
- Troubleshooting a Nonoperational VMware NSX Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN on page 147

Open vSwitch Database Schema For Physical Devices

Supported Platforms EX Series, MX Series, QFX Series standalone switches

An Open vSwitch Database (OVSDB) server runs on a Juniper Networks device that supports the OVSDB management protocol. When this device is connected to one or more VMware NSX controllers, the connections provide a means through which the Juniper Networks device and the controllers can communicate.

In an NSX multi-hypervisor environment, Juniper Networks devices that support OVSDB and NSX controllers exchange control and statistical data. This data is stored in the OVSDB database schema, which was defined for physical devices, and the schema resides in the OVSDB server. The schema includes several tables. Juniper Networks devices and NSX controllers, which are both OVSDB clients, can add rows to the tables as well as monitor for the addition, deletion, and modification of rows.

For example, the OVSDB client on a Juniper Networks device or NSX controller can collect MAC routes learned by entities in the physical or virtual networks, respectively, and publish the routes to the appropriate table in the schema. By using the MAC routes and other information provided in the table, Juniper Networks devices in the physical network and entities in the virtual network can determine where to forward virtual machine (VM) traffic.

Some of the OVSDB table names include the words *local* or *remote*, for example, the *unicast MACs local table* and the *unicast MACs remote table*. Information in *local* tables is learned by a Juniper Networks device that functions as a hardware virtual tunnel endpoint (VTEP), while information in *remote* tables is learned from other software or hardware VTEPs.

Table 6 on page 13 describes the tables in the schema, the physical or virtual entity that is the source of the data provided in the table, and the command that you can enter in the CLI of the Juniper Networks device to get similar information.

Table 6: OVSDB Schema Tables

Table Name	Description	Source of Information	Command
Global table	Includes the top-level configuration for the Juniper Networks device.	Juniper Networks device	None
Manager table	Includes information for each NSX controller that is connected to the Juniper Networks device.	• Juniper Networks device • NSX controller	show ovsdb controller
Physical switch table	Includes information about the Juniper Networks device on which a hardware VTEP is implemented. This table includes information only for the device on which the table resides.	Juniper Networks device	None
Physical port table	Includes information about OVSDB-managed interfaces.	Juniper Networks device	show ovsdb interface
Logical switch table	Includes information about logical switches, which you configured in NSX Manager or in the NSX API, and the corresponding Virtual Extensible LANs (VXLANs), which was configured on the Juniper Networks device.	Juniper Networks device	show ovsdb logical-switch
Logical binding statistics table	Includes statistics for OVSDB-managed interfaces.	Juniper Networks device	show ovsdb statistics interface
Physical locator table	Includes information about Juniper Networks devices configured as hardware VTEPs, software VTEPs, and service nodes.	Juniper Networks device	show ovsdb virtual-tunnel-end-point
Physical locator set table	Includes a list of service nodes for a logical switch.	Juniper Networks device	None
Unicast MACs remote table	Reachability information, including unicast MAC addresses, for entities in the virtual network.	NSX controller	show ovsdb mac
Unicast MACs local table	Reachability information, including unicast MAC addresses, for entities in the physical network.	Juniper Networks device that is configured as a hardware VTEP.	show ovsdb mac

Table 6: OVSDB Schema Tables *(continued)*

Table Name	Description	Source of Information	Command
Multicast MACs remote table	Includes only one row. In this row, the MAC column includes the keyword **unknown dst** along with a list of software VTEPs that host a cluster of service nodes, which handle multicast traffic.	NSX controller	show ovsdb mac

Related Documentation

- Understanding the Open vSwitch Database Management Protocol Running on Juniper Networks Devices on page 6
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7
- Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment on page 10

CHAPTER 2

VXLAN Overview

Understanding VXLANs

Supported Platforms EX Series, MX Series, QFX Series standalone switches

VXLAN Benefits

Virtual Extensible Local Area Network (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. 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 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.
  - QFX 5100 switches support 4K VXLANs.
- 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 don't need to use 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?

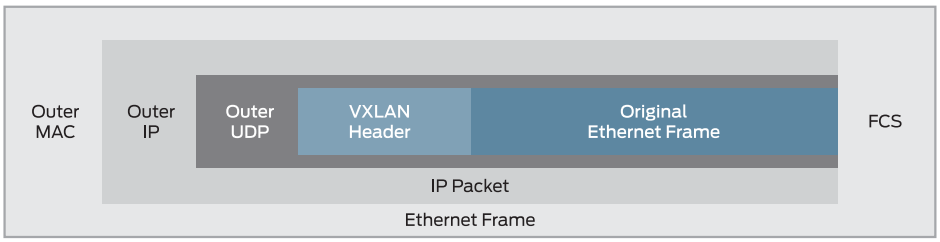
What is a VXLAN?

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. The devices that encapsulate traffic that must be transported over a VXLAN and de-encapsulate traffic when it must leave the VXLAN tunnel are virtual tunnel endpoints (VTEPs), which can be end hosts or network switches or routers. To encapsulate an Ethernet frame, VTEPs add a number of fields, including the following:

- Outer 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.

Figure 3 on page 16 shows the VXLAN packet format.

Figure 3: VXLAN Packet Format



Using a QFX5100 Switch with VXLANs

You can configure a QFX5100 switch to perform all of the following roles:

- 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 17 for more information.)
- Act as a Layer 2 gateway between virtualized and non-virtualized networks in the same data center or between data centers. For example, you can use a QFX5100 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 QFX5100 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.

NOTE: A QFX 5100 switch cannot route traffic between different VXLANs. To connect devices in different VXLANs you need a VXLAN-capable Layer 3 gateway, such as a Juniper Networks MX Series router.

Using an MX Series Routers as a VTEP

You can configure an MX Series router to act as a VTEP and perform all of the following roles:

- Act as a Layer 2 gateway between virtualized and non-virtualized 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 VPLS tunnels.

NOTE: If you want an MX Series router 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

You can use a controller (such as VMware's NSX) to 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 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.) 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 the appropriate Layer 2 interfaces. The remote VTEPs also do not copy and send copies of the packets according to the multicast tree.

Load Balancing VXLAN Traffic

On QFX5100 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.)

Related Documentation

- *Examples: Configuring VXLANs on QFX Series Switches*
- Example: Configuring VXLAN on MX Series Routers on page 45
- Open vSwitch Database Support on Juniper Networks Devices on page 3

PART 2

Configuration

CHAPTER 3

Configuration Examples

Example: Setting Up Inter-VXLAN Unicast Routing and OVSDB Connections in a Data Center

Supported Platforms EX Series, MX Series

This example shows how to set up a data center in which virtual machines (VMs) in different Virtual Extensible LANs (VXLANs) need to communicate. The Juniper Networks device that is integrated into this environment functions as a hardware virtual tunnel endpoint (VTEP) that can route VM traffic from one VXLAN (Layer 2) environment to another.

The Juniper Networks device implements the Open vSwitch Database (OVSDB) management protocol and has a connection with a VMware NSX controller, both of which enable the device and the NSX controller to exchange MAC routes to and from VMs in the physical and virtual networks.

This example explains how to configure a Juniper Networks device as a hardware VTEP, set up the routing of unicast packets between VXLANs, and set up an OVSDB connection with an NSX controller. For information about setting up the routing of unicast and multicast packets between VXLANs, see "Example: Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center" on page 32.

Requirements

The topology for this example includes the following hardware and software components:

- A cluster of five NSX controllers.
- NSX Manager.
- A service node that handles broadcast, unknown unicast, and multicast (BUM) traffic within each of the two VXLANs.
- Two hosts, each of which includes VMs managed by a hypervisor. Each hypervisor includes a software VTEP. The VMs on each of the hosts belong to different VXLANs.
- A Juniper Networks device that routes VM traffic between the two VXLANs. For example, an MX Series router running Junos OS Release 14.1R2 or later, or an EX9200 switch running Junos OS release 14.2 or later. The Juniper Networks device must also run an OVSDB software package, and the release of this package must be the same as the Junos OS release running on the device. This device is configured to function as a hardware VTEP.

Before you begin the configuration of the Juniper Networks device, you need to perform the following tasks:

- In NSX Manager or the NSX API, specify the IP address of the service node.
- In NSX Manager or the NSX API, configure a logical switch for each VXLAN that OVSDB will manage. This example implements two OVSDB-managed VXLANs; therefore, you must configure two logical switches. After the configuration of each logical switch, NSX automatically generates a universally unique identifier (UUID) for the logical switch. If you have not already, retrieve the UUID for each logical switch. A sample UUID is 28805c1d-0122-495d-85df-19abd647d772. When configuring the equivalent VXLANs on the Juniper Networks device, you must use the UUID of the logical switch as the bridge domain or VLAN name.

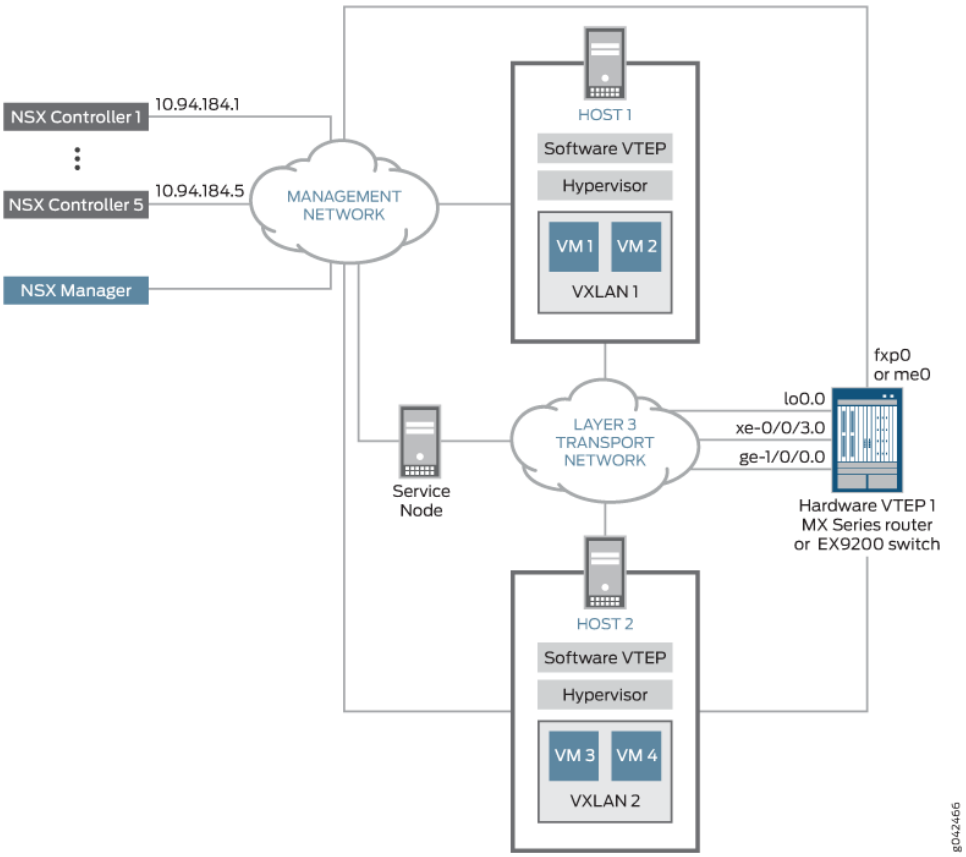
  For more information about logical switches and VXLANs, see "Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment" on page 10.
- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the Juniper Networks device. For more information, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers" on page 80.

For information about using NSX Manager or the NSX API to perform these configuration tasks, see the documentation that accompanies the respective products.

Overview and Topology

In the topology shown in Figure 4 on page 23, VM 1 in VXLAN 1 needs to communicate with VM 3 in VXLAN 2. To enable this communication, hardware VTEP 1, which can be an MX Series router or an EX9200 switch, is configured to route VM unicast traffic between the two VXLANs.

Figure 4: Inter-VXLAN Unicast Routing and OVSDB Topology



On hardware VTEP 1, a routing instance (virtual switch) is set up. Within the routing instance, two VXLANs are configured: VXLAN 1 and VXLAN 2. Each VXLAN has an integrated routing and bridging (IRB) interface associated with it. The IRB interfaces handle the routing of VM unicast traffic between the VXLANs,

Within each of the two VXLANs, a service node replicates Layer 2 BUM packets then forwards the replicas to all interfaces in the VXLANs. Having the service node handle the Layer 2 BUM traffic is the default behavior, and no configuration is required for this Juniper Networks device.

On hardware VTEP 1, a connection with an NSX controller is configured on the management interface (fxp0 for an MX Series router and me0 for an EX9200 switch). This configuration enables the NSX controller to push MAC routes for VM 1 and VM 3 to the hardware VTEP by way of the table for remote unicast MAC addresses in the OVSDB schema for physical devices.

Each VXLAN-encapsulated packet must include a source IP address, which identifies the source hardware or software VTEP, in the outer IP header. In this example, for hardware VTEP 1, the IP address of the loopback interface (lo0.0) is used.

In this example, the tracing of all OVSDB events is configured. The output of the OVSDB events are placed in a file named **ovsdb**, which is stored in the **/var/log** directory. By default, a maximum of 10 trace files can exist, and the configured maximum size of each file is 50 MB.

Table 7 on page 24 describes the components for setting up inter-VXLAN routing and an OVSDB connection.

Table 7: Components for Setting Up Inter-VXLAN Routing and OVSDB Connections in a Data Center

Property	Settings
Routing instance	Name: vx1 Type: virtual switch OVSDB-managed VXLANs included: VXLAN 1 and VXLAN 2
VXLAN 1	Bridge domain or VLAN associated with: 28805c1d-0122-495d-85df-19abd647d772 Interface: xe-0/0/2.0 VLAN ID: 100 VNI: 100
VXLAN 2	Bridge domain or VLAN associated with: 96a382cd-a570-4ac8-a77a-8bb8b16bde70 Interface: xe-1/2/0.0 VLAN ID: 200 VNI: 200
Inter-VXLAN unicast routing and forwarding with IRB interfaces	VXLAN 1: irb.0; 10.20.20.1/24; associated with routing interface vx1, and bridge domain or VLAN 28805c1d-0122-495d-85df-19abd647d772 VXLAN 2: irb.1; 10.10.10.3/24; associated with routing interface vx1, and bridge domain or VLAN 96a382cd-a570-4ac8-a77a-8bb8b16bde70
Handling of BUM traffic in each VXLAN	Service node NOTE: By default, one or more service nodes handle Layer 2 BUM traffic in a VXLAN; therefore, no configuration is required.
NSX controller	IP address: 10.94.184.1
Hardware VTEP source identifier	Source interface: loopback (lo0.0) Source IP address: 10.19.19.19/32

Table 7: Components for Setting Up Inter-VXLAN Routing and OVSDB Connections in a Data Center *(continued)*

Property	Settings
OVSDB tracing operations	Filename: /var/log/ovsdb File size: 50 MB Flag: All

Configuration

An MX Series router or an EX9200 switch can function as hardware VTEP 1 in this example. Because the configuration for each device is slightly different, a separate configuration is provided for each device.

To configure inter-VXLAN unicast routing and OVSDB connections in a data center topology, you need to perform one of these tasks:

- Configuring an MX Series Router as a Hardware VTEP with an OVSDB Connection on page 27
- Configuring an EX9200 Switch as a Hardware VTEP with an OVSDB Connection on page 29

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 (for example, IP addresses, interface names, and UUIDs), copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

MX Series router configuration:

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 unit 0 family inet address 10.50.50.2/24
set interfaces ge-1/0/0 unit 0 family inet address 10.100.100.99/24
set routing-options router-id 10.19.19.19
set protocols ospf area 0.0.0.0 interface xe-0/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set interfaces xe-0/0/2 unit 0 family bridge interface-mode access
set interfaces xe-0/0/2 unit 0 family bridge vlan-id 100
set interfaces xe-1/2/0 unit 0 family bridge interface-mode access
```

```
set interfaces xe-1/2/0 unit 0 family bridge vlan-id 200
set interfaces irb unit 0 family inet address 10.20.20.1/24
set interfaces irb unit 1 family inet address 10.10.10.3/24
set routing-instances vx1 vtep-source-interface lo0.0
set routing-instances vx1 instance-type virtual-switch
set routing-instances vx1 interface xe-0/0/2.0
set routing-instances vx1 interface xe-1/2/0.0
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vlan-id 100
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan ovsdb-managed
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan vni 100
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vlan-id 200
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan ovsdb-managed
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan vni 200
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 primary
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 preferred
set protocols ovsdb traceoptions file ovsdb
set protocols ovsdb traceoptions file size 50m
set protocols ovsdb traceoptions flag all
set protocols ovsdb controller 10.94.184.1
set protocols ovsdb interfaces xe-0/0/2.0
set protocols ovsdb interfaces xe-1/2/0.0
```

EX9200 switch configuration:

```
set interfaces xe-0/0/3 unit 0 family inet address 10.50.50.2/24
set interfaces ge-1/0/0 unit 0 family inet address 10.100.100.99/24
set routing-options router-id 10.19.19.19
set protocols ospf area 0.0.0.0 interface xe-0/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode access
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan-id 100
set interfaces xe-1/2/0 unit 0 family ethernet-switching interface-mode access
set interfaces xe-1/2/0 unit 0 family ethernet-switching vlan-id 200
set interfaces irb unit 0 family inet address 10.20.20.1/24
set interfaces irb unit 1 family inet address 10.10.10.3/24
set routing-instances vx1 vtep-source-interface lo0.0
set routing-instances vx1 instance-type virtual-switch
set routing-instances vx1 interface xe-0/0/2.0
set routing-instances vx1 interface xe-1/2/0.0
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vlan-id 100
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan
  ovsdb-managed
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni 100
```



```
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vlan-id 200
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan
  ovsdb-managed
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan vni
  200
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 preferred
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 primary
set protocols ovsdb traceoptions file ovsdb
set protocols ovsdb traceoptions file size 50m
set protocols ovsdb traceoptions flag all
set protocols ovsdb controller 10.94.184.1
set protocols ovsdb interfaces xe-0/0/2.0
set protocols ovsdb interfaces xe-1/2/0.0
```

Configuring an MX Series Router as a Hardware VTEP with an OVSDB Connection

Step-by-Step Procedure

To configure an MX Series router as hardware VTEP 1 with an OVSDB connection to an NSX controller, follow these steps:

1. Create the Layer 3 network.

```
[edit chassis]
user@router# set network-services enhanced-ip
[edit interfaces]
user@router# set xe-0/0/3 unit 0 family inet address 10.50.50.2/24
user@router# set ge-1/0/0 unit 0 family inet address 10.100.100.99/24
[edit routing-options]
user@router# set router-id 10.19.19.19
[edit protocols]
user@router# set ospf area 0.0.0.0 interface xe-0/0/3.0
user@router# set ospf area 0.0.0.0 interface ge-1/0/0.0
user@router# set ospf area 0.0.0.0 interface lo0.0
```

2. Create an access interface for VXLAN 1, and associate the interface with the VXLAN.

```
[edit interfaces]
user@router# set xe-0/0/2 unit 0 family bridge interface-mode access
user@router# set xe-0/0/2 unit 0 family bridge vlan-id 100
```

3. Create an access interface for VXLAN 2, and associate the interface with the VXLAN.

```
[edit interfaces]
user@router# set xe-1/2/0 unit 0 family bridge interface-mode access
user@router# set xe-1/2/0 unit 0 family bridge vlan-id 200
```

4. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 1.

```
[edit interfaces]
user@router# set irb unit 0 family inet address 10.20.20.1/24
```

5. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 2.

```
[edit interfaces]
user@router# set irb unit 1 family inet address 10.10.10.3/24
```

6. Set up the virtual switch routing instance.

```
[edit routing-instances]
user@router# set vx1 vtep-source-interface lo0.0
user@router# set vx1 instance-type virtual-switch
user@router# set vx1 interface xe-0/0/2.0
user@router# set vx1 interface xe-1/2/0.0
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vlan-id 100
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan ovsdb-managed
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan vni 100
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vlan-id 200
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan ovsdb-managed
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan vni 200
```

7. 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@router# set lo0 unit 0 family inet address 10.19.19.19/32 primary
user@router# set lo0 unit 0 family inet address 10.19.19.19/32 preferred
```

8. Set up OVSDB tracing operations.

```
[edit protocols]
user@router# set ovsdb traceoptions file ovsdb
user@router# set ovsdb traceoptions file size 50m
user@router# set ovsdb traceoptions flag all
```

9. Configure a connection with an NSX controller.

```
[edit protocols]
user@router# set ovsdb controller 10.94.184.1
```

10. Configure interfaces xe-0/0/2.0 and xe-1/2/0.0 to be managed by OVSDB.

```
[edit protocols]
user@router# set ovsdb interfaces xe-0/0/2.0
user@router# set ovsdb interfaces xe-1/2/0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port by using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

Configuring an EX9200 Switch as a Hardware VTEP with an OVSDB Connection

Step-by-Step Procedure

To configure an EX9200 switch as hardware VTEP 1 with an OVSDB connection to an NSX controller, follow these steps:

1. Create the Layer 3 network.

```
[edit chassis]
[edit interfaces]
user@switch# set xe-0/0/3 unit 0 family inet address 10.50.50.2/24
user@switch# set ge-1/0/0 unit 0 family inet address 10.100.100.99/24
[edit routing-options]
user@switch# set router-id 10.19.19.19
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface xe-0/0/3.0
user@switch# set ospf area 0.0.0.0 interface ge-1/0/0.0
user@switch# set ospf area 0.0.0.0 interface lo0.0
```

2. Create an access interface for VXLAN 1, and associate the interface with the VXLAN.

```
[edit interfaces]
user@switch# set xe-0/0/2 unit 0 family ethernet-switching interface-mode access
user@switch# set xe-0/0/2 unit 0 family ethernet-switching vlan-id 100
```

3. Create an access interface for VXLAN 2, and associate the interface with the VXLAN.

```
[edit interfaces]
user@switch# set xe-1/2/0 unit 0 family ethernet-switching interface-mode access
user@switch# set xe-1/2/0 unit 0 family ethernet-switching vlan-id 200
```

4. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 1.

```
[edit interfaces]
user@switch# set irb unit 0 family inet address 10.20.20.1/24
```

5. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 2.

```
[edit interfaces]
user@switch# set irb unit 1 family inet address 10.10.10.3/24
```

6. Set up the virtual switch routing instance.

```
[edit routing-instances]
user@switch# set vx1 vtep-source-interface lo0.0
user@switch# set vx1 instance-type virtual-switch
```

```
user@switch# set vx1 interface xe-0/0/2.0
user@switch# set vx1 interface xe-1/2/0.0
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vlan-id 100
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan
  ovsdb-managed
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni
  100
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vlan-id
  200
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan
  ovsdb-managed
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan vni
  200
```

7. 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 10.19.19.19/32 primary
user@switch# set lo0 unit 0 family inet address 10.19.19.19/32 preferred
```

8. Set up tracing operations to be performed for the OVSDB management protocol.

```
[edit protocols]
user@switch# set ovsdb traceoptions file ovsdb
user@switch# set ovsdb traceoptions file size 50m
user@switch# set ovsdb traceoptions flag all
```

9. Configure a connection with an NSX controller.

```
[edit protocols]
user@switch# set ovsdb controller 10.94.184.1
```

10. Configure interfaces xe-0/0/2.0 and xe-1/2/0.0 to be managed by OVSDB.

```
[edit protocols]
user@router# set ovsdb interfaces xe-0/0/2.0
user@router# set ovsdb interfaces xe-1/2/0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port by using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

Verification

Verifying the Logical Switches

Purpose Verify that logical switches with the UUIDs of 28805c1d-0122-495d-85df-19abd647d772 and 96a382cd-a570-4ac8-a77a-8bb8b16bde70 are configured in NSX Manager or in the NSX API, and that information about the logical switches is published in the OVSDB schema.

Action Issue the **show ovsdb logical-switch** operational mode command.

```
user@host> show ovsdb logical-switch
Logical switch information:
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
Flags: Created by both
VNI: 100
Num of Remote MAC: 1
Num of Local MAC: 0
Logical Switch Name: 96a382cd-a570-4ac8-a77a-8bb8b16bde70
Flags: Created by both
VNI: 200
Num of Remote MAC: 1
Num of Local MAC: 1
```

Meaning The output verifies that information about the logical switches is published in the OVSDB schema. The **Created by both** state indicates that the logical switches are configured in NSX Manager or the NSX API, and the corresponding VXLANs are configured on the Juniper Networks device. In this state, the logical switches and VXLANs are operational.

If the state of the logical switches is something other than **Created by both**, see "Troubleshooting a Nonoperational VMware NSX Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN" on page 147.

Verifying the MAC Addresses of VM 1 and VM 3

Purpose Verify that the MAC addresses of VM 1 and VM 3 are present in the OVSDB schema.

Action Issue the **show ovsdb mac remote** operational mode command to verify that the MAC addresses for VM 1 and VM 3 are present.

```
user@host> show ovsdb mac remote
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
  Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  08:33:9d:5f:a7:f1     0.0.0.0           Vxlan over Ipv4    10.19.19.19
Logical Switch Name: 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  a8:59:5e:f6:38:90     0.0.0.0           Vxlan over Ipv4    10.19.19.10
```

Meaning The output shows that the MAC addresses for VM 1 and VM 3 are present and are associated with logical switches with the UUIDs of **28805c1d-0122-495d-85df-19abd647d772** and **96a382cd-a570-4ac8-a77a-8bb8b16bde70**, respectively. Given that the MAC addresses are present, VM 1 and VM 3 are reachable through hardware VTEP 1.

Verifying the NSX Controller Connection

Purpose Verify that the connection with the NSX controller is up.

Action Issue the **show ovsdb controller** operational mode command, and verify that the controller connection state is **up**.

```
user@host> show ovsdb controller
VTEP controller information:
Controller IP address: 10.94.184.1
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 542325
Controller seconds-since-disconnect: 542346
Controller connection status: active
```

Meaning The output shows that the connection state of the NSX controller is **up**, in addition to other information about the controller. When this connection is up, OVSDB is enabled on the Juniper Networks device.

Related Documentation

- Open vSwitch Database Schema For Physical Devices on page 12

Example: Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center

Supported Platforms EX Series, MX Series

This example shows how to set up a data center in which virtual machines (VMs) in different Virtual Extensible LANs (VXLANs) need to communicate. The Juniper Networks device that is integrated into this environment functions as a hardware virtual tunnel

endpoint (VTEP) that can route VM traffic from one VXLAN (Layer 2) environment to another.

The Juniper Networks device implements the Open vSwitch Database (OVSDB) management protocol and has a connection with a VMware NSX controller, both of which enable the device and the NSX controller to exchange MAC routes to and from VMs in the physical and virtual networks.

This example explains how to configure a Juniper Networks device as a hardware VTEP, set up the routing of unicast and multicast packets between VXLANs, and set up an OVSDB connection with an NSX controller. For information about setting up the routing of unicast packets only between VXLANs, see "Example: Setting Up Inter-VXLAN Unicast Routing and OVSDB Connections in a Data Center" on page 21.

Requirements

The topology for this example includes the following hardware and software components:

- A cluster of five NSX controllers.
- NSX Manager.
- A service node that handles broadcast, unknown unicast, and multicast (BUM) traffic within each of the two VXLANs.
- Two hosts, each of which includes VMs managed by a hypervisor. Each hypervisor includes a software VTEP. The VMs on each of the hosts belong to different VXLANs.
- A Juniper Networks device that routes VM traffic between the two VXLANs. For example, an MX Series router running Junos OS Release 14.1R2 or later, or an EX9200 switch running Junos OS release 14.2 or later. The Juniper Networks device must also run an OVSDB software package, and the release of this package must be the same as the Junos OS release running on the device. This device is configured to function as a hardware VTEP.

Before you begin the configuration of the Juniper Networks device, you need to perform the following tasks:

- In NSX Manager or the NSX API, specify the IP address of the service node.
- In NSX Manager or the NSX API, configure a logical switch for each VXLAN that OVSDB will manage. This example implements two OVSDB-managed VXLANs; therefore, you must configure two logical switches. After the configuration of each logical switch, NSX automatically generates a universally unique identifier (UUID) for the logical switch. If you have not already, retrieve the UUID for each logical switch. A sample UUID is 28805c1d-0122-495d-85df-19abd647d772. When configuring the equivalent VXLANs on the Juniper Networks device, you must use the UUID of the logical switch as the bridge domain or VLAN name.

For more information about logical switches and VXLANs, see "Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment" on page 10.

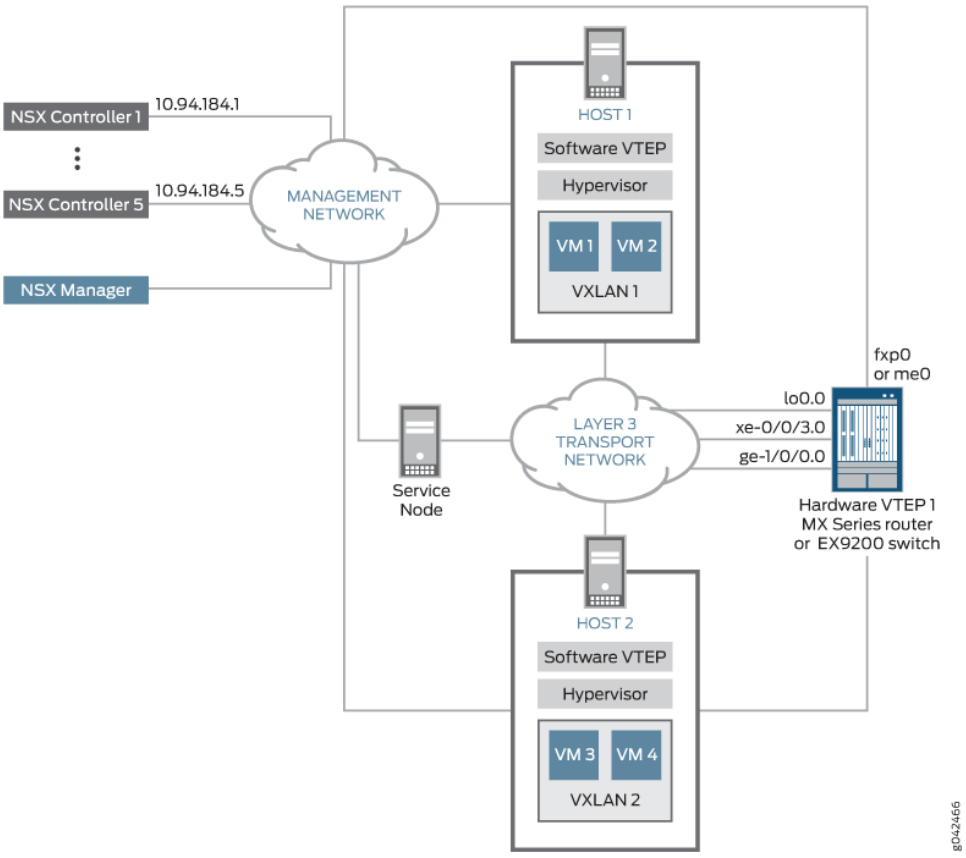
- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the Juniper Networks device. For more information, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers" on page 80.

For information about using NSX Manager or the NSX API to perform these configuration tasks, see the documentation that accompanies the respective products.

Overview and Topology

In the topology shown in Figure 4 on page 23, VM 1 in VXLAN 1 needs to communicate with VM 3 in VXLAN 2. To enable this communication, hardware VTEP 1, which can be an MX Series router or an EX9200 switch, is configured to route VM traffic between the two VXLANs.

Figure 5: Inter-VXLAN Unicast and Multicast Routing and OVSDB Topology



On hardware VTEP 1, a routing instance (virtual switch) is set up. Within the routing instance, two VXLANs are configured: VXLAN 1 and VXLAN 2. Each VXLAN has an

integrated routing and bridging (IRB) interface associated with it. The IRB interfaces handle the routing of VM unicast traffic between the VXLANs,

To handle multicast traffic between the VXLANs, each IRB interface is configured as a member of an Internet Group Management Protocol (IGMP) static group, and the MX Series router or EX9200 switch is configured to function as a PIM rendezvous point (RP) that forwards multicast traffic to each VXLAN through its associated IRB interface.

In this topology, when a multicast packet is received by a VXLAN, for example, VXLAN 1, the following packet handling occurs:

- Within VXLAN 1, the packet is handled as a Layer 2 multicast packet, which means that it is sent to the service node. The service node replicates Layer 2 multicast, as well as Layer 2 broadcast and unknown unicast, packets then forwards the replicas to all interfaces in VXLAN 1. Having the service node handle the Layer 2 BUM traffic is the default behavior, and no configuration is required for the MX Series router or the EX9200 switch.
- The IRB interface associated with VXLAN 1 sends the packet to the PIM RP, which forwards the packet to the IRB associated with VXLAN 2. The IRB interface associated with VXLAN 2 then replicates the packet and forwards the replicas to all hardware and software VTEPs that host VMs, but not to service nodes, in VXLAN 2. The ability of an IRB interface to replicate the Layer 3 multicast packets and forward the replicas to hardware and software VTEPs in a VXLAN is known as *ingress node replication*. This feature is automatically implemented and does not need to be configured.

On hardware VTEP 1, a connection with an NSX controller is configured on the management interface (fxp0 for an MX Series router and me0 for an EX9200 switch). This configuration enables the NSX controller to push MAC routes for VM 1 and VM 3 to the hardware VTEP by way of the table for remote unicast MAC addresses in the OVSDB schema for physical devices.

Each VXLAN-encapsulated packet must include a source IP address, which identifies the source hardware or software VTEP, in the outer IP header. In this example, for hardware VTEP 1, the IP address of the loopback interface (lo0.0) is used.

In this example, the tracing of all OVSDB events is configured. The output of the OVSDB events are placed in a file named **ovsdb**, which is stored in the **/var/log** directory. By default, a maximum of 10 trace files can exist, and the configured maximum size of each file is 50 MB.

Table 7 on page 24 describes the components for setting up inter-VXLAN routing and an OVSDB connection.

Table 8: Components for Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center

Property	Settings
Routing instance	Name: vx1 Type: virtual switch OVSDB-managed VXLANs included: VXLAN 1 and VXLAN 2
VXLAN 1	Bridge domain or VLAN associated with: 28805c1d-0122-495d-85df-19abd647d772 Interface: xe-0/0/2.0 VLAN ID: 100 VNI: 100
VXLAN 2	Bridge domain or VLAN associated with: 96a382cd-a570-4ac8-a77a-8bb8b16bde70 Interface: xe-1/2/0.0 VLAN ID: 200 VNI: 200
Inter-VXLAN unicast routing and forwarding with IRB interfaces	VXLAN 1: irb.0; 10.20.20.1/24; associated with routing interface vx1, and bridge domain or VLAN 28805c1d-0122-495d-85df-19abd647d772 VXLAN 2: irb.1; 10.10.10.3/24; associated with routing interface vx1, and bridge domain or VLAN 96a382cd-a570-4ac8-a77a-8bb8b16bde70
Inter-VXLAN multicast routing and forwarding with IRB interfaces	PIM RP: 10.19.19.19 VXLAN 1: PIM interface irb.0; IGMP static group 225.1.1.100 VXLAN 2: PIM interface irb.1; IGMP static groups 225.1.1.100 NOTE: On IRB interfaces that forward Layer 3 multicast traffic from one OVSDB-managed VXLAN to another, ingress node replication is automatically implemented; therefore, no configuration is required.
Handling of Layer 2 BUM traffic in each VXLAN	Service node NOTE: By default, one or more service nodes handle Layer 2 BUM traffic in a VXLAN; therefore, no configuration is required.
NSX controller	IP address: 10.94.184.1
Hardware VTEP source identifier	Source interface: loopback (lo0.0) Source IP address: 10.19.19.19/32

Table 8: Components for Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center *(continued)*

Property	Settings
OVSDB tracing operations	Filename: /var/log/ovsdb File size: 50 MB Flag: All

Configuration

An MX Series router or an EX9200 switch can function as hardware VTEP 1 in this example. Because the configuration for each device is slightly different, a separate configuration is provided for each device.

To configure inter-VXLAN unicast and multicast routing and OVSDB connections in a data center topology, you need to perform one of these tasks:

- Configuring an MX Series Router as a Hardware VTEP with an OVSDB Connection on page 39
- Configuring an EX9200 Switch as a Hardware VTEP with an OVSDB Connection on page 41

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 (for example, IP addresses, interface names, and UUIDs), copy and paste the commands into the CLI at the [**edit**] hierarchy level, and then enter **commit** from configuration mode.

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

MX Series router configuration:

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/3 unit 0 family inet address 10.50.50.2/24
set interfaces ge-1/0/0 unit 0 family inet address 10.100.100.99/24
set routing-options router-id 10.19.19.19
set protocols ospf area 0.0.0.0 interface xe-0/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set interfaces xe-0/0/2 unit 0 family bridge interface-mode access
set interfaces xe-0/0/2 unit 0 family bridge vlan-id 100
set interfaces xe-1/2/0 unit 0 family bridge interface-mode access
```

```
set interfaces xe-1/2/0 unit 0 family bridge vlan-id 200
set interfaces irb unit 0 family inet address 10.20.20.1/24
set interfaces irb unit 1 family inet address 10.10.10.3/24
set protocols igmp interface irb.0 static group 225.1.1.100
set protocols igmp interface irb.1 static group 225.1.1.100
set protocols pim rp local address 10.19.19.19
set protocols pim interface irb.0
set protocols pim interface irb.1
set routing-instances vx1 vtep-source-interface lo0.0
set routing-instances vx1 instance-type virtual-switch
set routing-instances vx1 interface xe-0/0/2.0
set routing-instances vx1 interface xe-1/2/0.0
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vlan-id 100
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan ovsdb-managed
set routing-instances vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan vni 100
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vlan-id 200
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan ovsdb-managed
set routing-instances vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan vni 200
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 primary
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 preferred
set protocols ovsdb traceoptions file ovsdb
set protocols ovsdb traceoptions file size 50m
set protocols ovsdb traceoptions flag all
set protocols ovsdb controller 10.94.184.1
set protocols ovsdb interfaces xe-0/0/2.0
set protocols ovsdb interfaces xe-1/2/0.0
```

EX9200 switch configuration:

```
set interfaces xe-0/0/3 unit 0 family inet address 10.50.50.2/24
set interfaces ge-1/0/0 unit 0 family inet address 10.100.100.99/24
set routing-options router-id 10.19.19.19
set protocols ospf area 0.0.0.0 interface xe-0/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set interfaces xe-0/0/2 unit 0 family ethernet-switching interface-mode access
set interfaces xe-0/0/2 unit 0 family ethernet-switching vlan-id 100
set interfaces xe-1/2/0 unit 0 family ethernet-switching interface-mode access
set interfaces xe-1/2/0 unit 0 family ethernet-switching vlan-id 200
set interfaces irb unit 0 family inet address 10.20.20.1/24
set interfaces irb unit 1 family inet address 10.10.10.3/24
set protocols igmp interface irb.0 static group 225.1.1.100
set protocols igmp interface irb.1 static group 225.1.1.100
set protocols pim rp local address 10.19.19.19
set protocols pim interface irb.0
set protocols pim interface irb.1
```

```
set routing-instances vx1 vtep-source-interface lo0.0
set routing-instances vx1 instance-type virtual-switch
set routing-instances vx1 interface xe-0/0/2.0
set routing-instances vx1 interface xe-1/2/0.0
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vlan-id 100
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan
  ovsdb-managed
set routing-instances vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni 100
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vlan-id 200
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan
  ovsdb-managed
set routing-instances vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan vni
  200
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 preferred
set interfaces lo0 unit 0 family inet address 10.19.19.19/32 primary
set protocols ovsdb traceoptions file ovsdb
set protocols ovsdb traceoptions file size 50m
set protocols ovsdb traceoptions flag all
set protocols ovsdb controller 10.94.184.1
set protocols ovsdb interfaces xe-0/0/2.0
set protocols ovsdb interfaces xe-1/2/0.0
```

Configuring an MX Series Router as a Hardware VTEP with an OVSDB Connection

Step-by-Step Procedure

To configure an MX Series router as hardware VTEP 1 with an OVSDB connection to an NSX controller, follow these steps:

1. Create the Layer 3 network.

```
[edit chassis]
user@router# set network-services enhanced-ip
[edit interfaces]
user@router# set xe-0/0/3 unit 0 family inet address 10.50.50.2/24
user@router# set ge-1/0/0 unit 0 family inet address 10.100.100.99/24
[edit routing-options]
user@router# set router-id 10.19.19.19
[edit protocols]
user@router# set ospf area 0.0.0.0 interface xe-0/0/3.0
user@router# set ospf area 0.0.0.0 interface ge-1/0/0.0
user@router# set ospf area 0.0.0.0 interface lo0.0
```

2. Create an access interface for VXLAN 1, and associate the interface with the VXLAN.

```
[edit interfaces]
user@router# set xe-0/0/2 unit 0 family bridge interface-mode access
user@router# set xe-0/0/2 unit 0 family bridge vlan-id 100
```

3. Create an access interface for VXLAN 2, and associate the interface with the VXLAN.

```
[edit interfaces]
user@router# set xe-1/2/0 unit 0 family bridge interface-mode access
user@router# set xe-1/2/0 unit 0 family bridge vlan-id 200
```

4. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 1.

```
[edit interfaces]
user@router# set irb unit 0 family inet address 10.20.20.1/24
```

5. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 2.

```
[edit interfaces]
user@router# set irb unit 1 family inet address 10.10.10.3/24
```

6. Configure PIM and IGMP to handle inter-VXLAN multicast traffic.

```
[edit protocols]
user@router# set pim rp local address 10.19.19.19
user@router# set pim interface irb.0
user@router# set pim interface irb.1
user@router# set igmp interface irb.0 static group 225.1.1.100
user@router# set igmp interface irb.1 static group 225.1.1.100
```

7. Set up the virtual switch routing instance.

```
[edit routing-instances]
user@router# set vx1 vtep-source-interface lo0.0
user@router# set vx1 instance-type virtual-switch
user@router# set vx1 interface xe-0/0/2.0
user@router# set vx1 interface xe-1/2/0.0
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vlan-id 100
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan ovsdb-managed
user@router# set vx1 bridge-domains 28805c1d-0122-495d-85df-19abd647d772
  vxlan vni 100
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vlan-id 200
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan ovsdb-managed
user@router# set vx1 bridge-domains 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  vxlan vni 200
```

8. 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@router# set lo0 unit 0 family inet address 10.19.19.19/32 primary
user@router# set lo0 unit 0 family inet address 10.19.19.19/32 preferred
```

9. Set up OVSDB tracing operations.

```
[edit protocols]
user@router# set ovsdb traceoptions file ovsdb
user@router# set ovsdb traceoptions file size 50m
user@router# set ovsdb traceoptions flag all
```

10. Configure a connection with an NSX controller.

```
[edit protocols]
user@router# set ovsdb controller 10.94.184.1
```

11. Configure interfaces xe-0/0/2.0 and xe-1/2/0.0 to be managed by OVSDB.

```
[edit protocols]
user@router# set ovsdb interfaces xe-0/0/2.0
user@router# set ovsdb interfaces xe-1/2/0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port by using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

Configuring an EX9200 Switch as a Hardware VTEP with an OVSDB Connection

Step-by-Step Procedure

To configure an EX9200 switch as hardware VTEP 1 with an OVSDB connection to an NSX controller, follow these steps:

1. Create the Layer 3 network.

```
[edit chassis]
[edit interfaces]
user@switch# set xe-0/0/3 unit 0 family inet address 10.50.50.2/24
user@switch# set ge-1/0/0 unit 0 family inet address 10.100.100.99/24
[edit routing-options]
user@switch# set router-id 10.19.19.19
[edit protocols]
user@switch# set ospf area 0.0.0.0 interface xe-0/0/3.0
user@switch# set ospf area 0.0.0.0 interface ge-1/0/0.0
user@switch# set ospf area 0.0.0.0 interface lo0.0
```

2. Create an access interface for VXLAN 1, and associate the interface with the VXLAN.

```
[edit interfaces]
user@switch# set xe-0/0/2 unit 0 family ethernet-switching interface-mode access
user@switch# set xe-0/0/2 unit 0 family ethernet-switching vlan-id 100
```

3. Create an access interface for VXLAN 2, and associate the interface with the VXLAN.

```
[edit interfaces]
user@switch# set xe-1/2/0 unit 0 family ethernet-switching interface-mode access
user@switch# set xe-1/2/0 unit 0 family ethernet-switching vlan-id 200
```

4. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 1.

```
[edit interfaces]
user@switch# set irb unit 0 family inet address 10.20.20.1/24
```

5. Create an IRB interface to handle inter-VXLAN unicast traffic for VXLAN 2.

```
[edit interfaces]
user@switch# set irb unit 1 family inet address 10.10.10.3/24
```

6. Configure PIM and IGMP to handle inter-VXLAN multicast traffic.

```
[edit protocols]
user@switch# set pim rp local address 10.19.19.19
user@switch# set pim interface irb.0
user@switch# set pim interface irb.1
user@switch# set igmp interface irb.0 static group 225.1.1.100
user@switch# set igmp interface irb.1 static group 225.1.1.100
```

7. Set up the virtual switch routing instance.

```
[edit routing-instances]
user@switch# set vx1 vtep-source-interface lo0.0
user@switch# set vx1 instance-type virtual-switch
user@switch# set vx1 interface xe-0/0/2.0
user@switch# set vx1 interface xe-1/2/0.0
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vlan-id 100
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772
  routing-interface irb.0
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan
  ovsdb-managed
user@switch# set vx1 vlans 28805c1d-0122-495d-85df-19abd647d772 vxlan vni
  100
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vlan-id
  200
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  routing-interface irb.1
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan
  ovsdb-managed
user@switch# set vx1 vlans 96a382cd-a570-4ac8-a77a-8bb8b16bde70 vxlan vni
  200
```

8. 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 10.19.19.19/32 primary
user@switch# set lo0 unit 0 family inet address 10.19.19.19/32 preferred
```

9. Set up tracing operations to be performed for the OVSDB management protocol.

```
[edit protocols]
user@switch# set ovsdb traceoptions file ovsdb
user@switch# set ovsdb traceoptions file size 50m
user@switch# set ovsdb traceoptions flag all
```

10. Configure a connection with an NSX controller.

```
[edit protocols]
user@switch# set ovsdb controller 10.94.184.1
```

11. Configure interfaces xe-0/0/2.0 and xe-1/2/0.0 to be managed by OVSDB.

```
[edit protocols]
```

```
user@router# set ovsdb interfaces xe-0/0/2.0
user@router# set ovsdb interfaces xe-1/2/0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX-equivalent of a hardware VTEP. This example implements one hardware VTEP; therefore, you must configure one gateway, a gateway service, and a logical switch port by using NSX Manager or the NSX API. For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

Verification

Verifying the Logical Switches

Purpose Verify that logical switches with the UUIDs of 28805c1d-0122-495d-85df-19abd647d772 and 96a382cd-a570-4ac8-a77a-8bb8b16bde70 are configured in NSX Manager or in the NSX API, and that information about the logical switches is published in the OVSDB schema.

Action Issue the **show ovsdb logical-switch** operational mode command.

```
user@host> show ovsdb logical-switch
Logical switch information:
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
Flags: Created by both
VNI: 100
Num of Remote MAC: 1
Num of Local MAC: 0
Logical Switch Name: 96a382cd-a570-4ac8-a77a-8bb8b16bde70
Flags: Created by both
VNI: 200
Num of Remote MAC: 1
Num of Local MAC: 1
```

Meaning The output verifies that information about the logical switches is published in the OVSDB schema. The **Created by both** state indicates that the logical switches are configured in NSX Manager or the NSX API, and the corresponding VXLANs are configured on the Juniper Networks device. In this state, the logical switches and VXLANs are operational.

If the state of the logical switches is something other than **Created by both**, see "Troubleshooting a Nonoperational VMware NSX Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN" on page 147.

Verifying the MAC Addresses of VM 1 and VM 3

Purpose Verify that the MAC addresses of VM 1 and VM 3 are present in the OVSDB schema.

Action Issue the **show ovsdb mac remote** operational mode command to verify that the MAC addresses for VM 1 and VM 3 are present.

```
user@host> show ovsdb mac remote
Logical Switch Name: 28805c1d-0122-495d-85df-19abd647d772
  Mac                  IP                 Encapsulation      Vtep
  Address              Address                               Address
  08:33:9d:5f:a7:f1    0.0.0.0            Vxlan over Ipv4    10.19.19.19
Logical Switch Name: 96a382cd-a570-4ac8-a77a-8bb8b16bde70
  Mac                  IP                 Encapsulation      Vtep
  Address              Address                               Address
  a8:59:5e:f6:38:90    0.0.0.0            Vxlan over Ipv4    10.19.19.10
```

Meaning The output shows that the MAC addresses for VM 1 and VM 3 are present and are associated with logical switches with the UUIDs of **28805c1d-0122-495d-85df-19abd647d772** and **96a382cd-a570-4ac8-a77a-8bb8b16bde70**, respectively. Given that the MAC addresses are present, VM 1 and VM 3 are reachable through hardware VTEP 1.

Verifying the NSX Controller Connection

Purpose Verify that the connection with the NSX controller is up.

Action Issue the **show ovsdb controller** operational mode command, and verify that the controller connection state is **up**.

```
user@host> show ovsdb controller
VTEP controller information:
Controller IP address: 10.94.184.1
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 542325
Controller seconds-since-disconnect: 542346
Controller connection status: active
```

Meaning The output shows that the connection state of the NSX controller is **up**, in addition to other information about the controller. When this connection is up, OVSDB is enabled on the Juniper Networks device.

Related Documentation

- Understanding How Layer 2 BUM Traffic and Layer 3 Routed Multicast Traffic Are Handled in VXLANs Managed by OVSDB on page 8
- Open vSwitch Database Schema For Physical Devices on page 12

Example: Configuring VXLAN on MX Series Routers

Supported Platforms MX Series

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). VTEPs encapsulate the virtual machine traffic into a VXLAN header and strip off the encapsulation.

This example shows how to configure VXLAN on MX Series routers using switch options in a default bridge domain.

Requirements

This example uses the following hardware and software components:

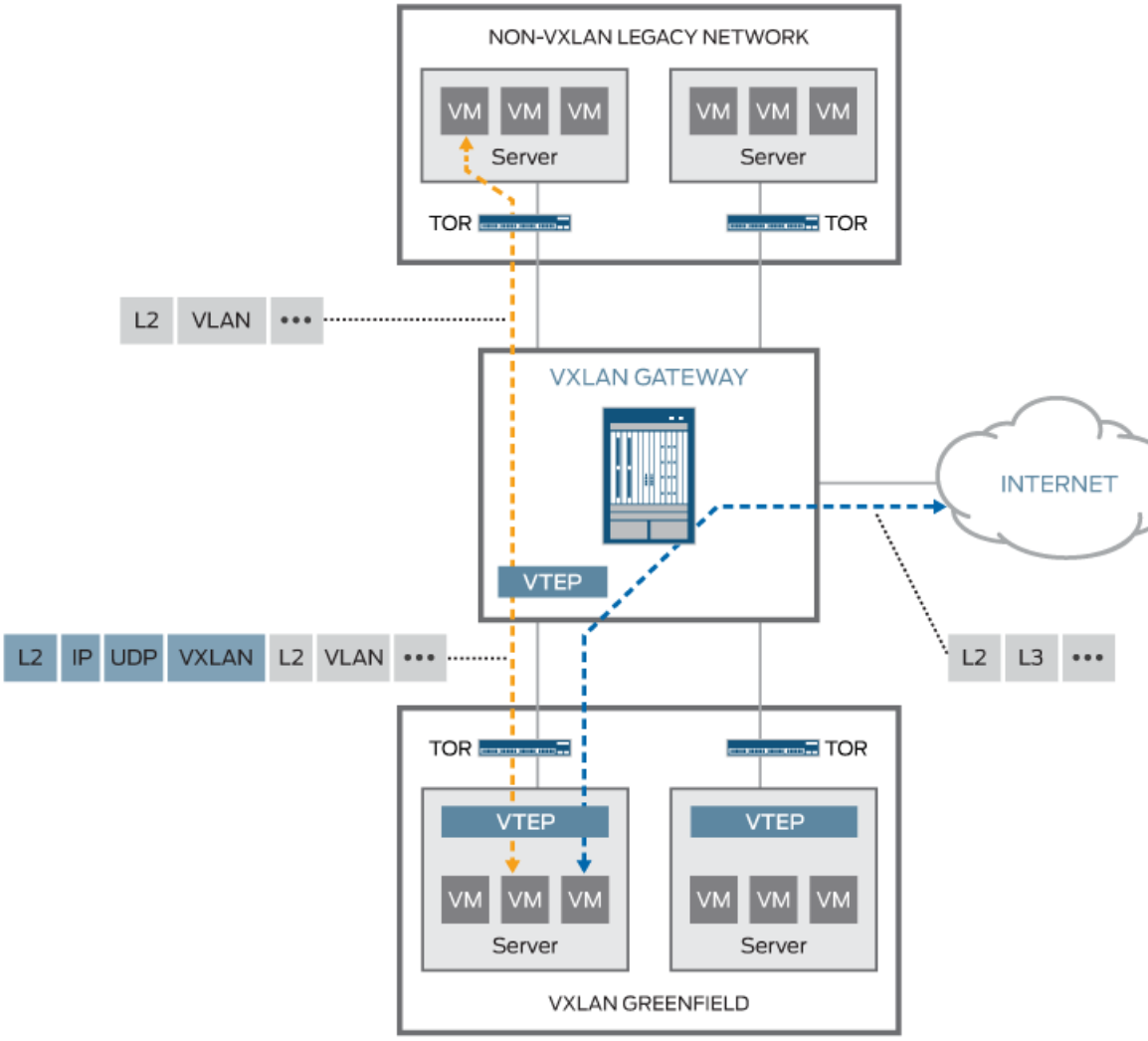
- An MX Series router
- A VXLAN capable peer router
- Junos OS Release 14.1

Overview

In this example, VXLAN is configured to run on a default bridge domain. VTEP interfaces sources are configured to the loopback address, and VLAN groups are configured under bridge domains with VXLAN enabled. Interfaces are configured for VLAN tagging and encapsulation, and IRB is enabled. OSPF and PIM protocols are configured to facilitate unicast and multicast routing. The chassis is configured for GRES and enhanced IP services.

Topology

Figure 1: VXLAN Topology



Configuring VXLAN on MX Series Routers

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.

```
set switch-options vtep-source-interface lo0.0
set bridge-domains vlan-5 vxlan vni 100
set bridge-domains vlan-5 vxlan multicast-group 239.1.1.1
set bridge-domains vlan-5 vlan-id 100
set bridge-domains vlan-5 routing-interface irb.0
set bridge-domains vlan-5 interface xe-1/0/0.0
```

```
set bridge-domains vlan-6 vxlan vni 200
set bridge-domains vlan-6 vxlan multicast-group 239.1.1.1
set bridge-domains vlan-6 vlan-id 200
set bridge-domains vlan-6 routing-interface irb.1
set bridge-domains vlan-6 interface xe-2/0/0.0
set interfaces xe-1/0/0 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 100
set interfaces xe-2/0/0 vlan-tagging
set interfaces xe-2/0/0 encapsulation flexible-ethernet-services
set interfaces xe-2/0/0 unit 0 encapsulation vlan-bridge
set interfaces xe-2/0/0 unit 0 vlan-id 200
set interface irb unit 0 family inet address 5.5.5.1/24
set interface irb unit 1 family inet address 6.6.6.1/24
set interfaces lo0 unit 0 family inet address 3.3.3.3/32
set protocols ospf area 0.0.0.0 interface ge-8/3/8.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface xe-0/1/3.0
set protocols ospf area 0.0.0.0 interface ge-8/3/2.0
set protocols pim rp static address 10.2.1.3
set protocols pim interface lo0.0 mode bidirectional-sparse
set protocols pim interface ge-8/3/8.0 mode bidirectional-sparse
set protocols pim interface xe-0/1/3.0 mode bidirectional-sparse
set protocols pim interface ge-8/3/2.0 mode bidirectional-sparse
set chassis redundancy graceful-switchover
set chassis aggregated-devices ethernet device-count 10
set chassis fpc 1 pic 0 tunnel-services bandwidth 10g
set chassis network-services enhanced-ip
```

Configuring VXLAN

Step-by-Step Procedure

The following example show how to set up a basic VXLAN configuration with default bridge domains and switch options. To configure VXLAN on an MX Series router, follow these steps:

1. Configure VTEP interface sources under **switch-options** for the default-switch.

   ```
   [edit]
   user@router# set switch-options vtep-source-interface lo0.0
   ```

2. Set up a VLAN group named **vlan-5** and set its VXLAN Network Identifier (VNI) to 100.

   ```
   [edit]
   user@router# set bridge-domains vlan-5 vxlan vni 100
   ```

3. Configure the **vlan-5** multicast group address for VXLAN.

   ```
   [edit]
   user@router# set bridge-domains vlan-5 vxlan multicast-group 239.1.1.1
   ```

4. Set the VLAN ID to 100 for **vlan-5**.

   ```
   [edit]
   user@router# set bridge-domains vlan-5 vlan-id 100
   ```

5. Configure integrated bridging and routing (IRB) for **vlan-5**.

```
[edit]
user@router# set bridge-domains vlan-5 routing-interface irb.0
```

6. Assign the xe-1/0/0.0 interface to **vlan-5**.

```
[edit]
user@router# set bridge-domains vlan-5 interface xe-1/0/0.0
```

7. Set up a VLAN group named **vlan-6** and set its VXLAN Network Identifier (VNI) to 200.

```
[edit]
user@router# set bridge-domains vlan-6 vxlan vni 200
```

8. Configure the **vlan-6** multicast group address for VXLAN.

```
[edit]
user@router# set bridge-domains vlan-6 vxlan multicast-group 239.1.1.1
```

9. Set the VLAN ID to 100 for **vlan-6**.

```
[edit]
user@router# set bridge-domains vlan-6 vlan-id 200
```

10. Configure IRB for **vlan-6**.

```
[edit]
user@router# set bridge-domains vlan-6 routing-interface irb.1
```

11. Assign the xe-2/0/0.0 interface to **vlan-6**.

```
[edit]
user@router# set bridge-domains vlan-6 interface xe-2/0/0.0
```

12. Set up VLAN tagging for xe-1/0/0.

```
[edit]
user@router# set interfaces xe-1/0/0 vlan-tagging
```

13. Configure flexible Ethernet service encapsulation on xe-1/0/0.

```
[edit]
user@router# set interfaces xe-1/0/0 encapsulation flexible-ethernet-services
```

14. Set up VLAN bridging encapsulation for xe-1/0/0 unit 0 ˙.

```
[edit]
user@router# set interfaces xe-1/0/0 unit 0 encapsulation vlan-bridge
```

15. Set the xe-1/0/0 unit 0 VLAN ID to 100.

```
[edit]
user@router# set interfaces xe-1/0/0 unit 0 vlan-id 100
```

16. Configure VLAN tagging for xe-2/0/0

```
[edit]
user@router# set interfaces xe-2/0/0 vlan-tagging
```

17. Set up flexible Ethernet service encapsulation on xe-2/0/0.

```
[edit]
user@router# set interfaces xe-2/0/0 encapsulation flexible-ethernet-services
```

18. Configure VLAN bridging encapsulation for xe-2/0/0 unit 0 .

```
[edit]
user@router# set interfaces xe-2/0/0 unit 0 encapsulation vlan-bridge
```

19. Set the xe-2/0/0 unit 0 VLAN ID to 200.

```
[edit]
user@router# set interfaces xe-2/0/0 unit 0 vlan-id 200
```

20. Configure the IRB unit 0 family inet address.

```
[edit]
user@router# set interface irb unit 0 family inet address 5.5.5.1/24
```

21. Configure the IRB unit 1 family inet address.

```
[edit]
user@router# set interface irb unit 1 family inet address 6.6.6.1/24
```

22. Set the family inet address for the loopback unit 0.

```
[edit]
user@router# set interfaces lo0 unit 0 family inet address 3.3.3.3/32
```

23. Set up OSPF for the ge-8/3/8.0 interface.

```
[edit]
user@router# set protocols ospf area 0.0.0.0 interface ge-8/3/8.0
```

24. Configure OSPF for the loopback interface.

```
[edit]
user@router# set protocols ospf area 0.0.0.0 interface lo0.0
```

25. Set up OSPF for the xe-0/1/3.0 interface.

```
[edit]
user@router# set protocols ospf area 0.0.0.0 interface xe-0/1/3.0
```

26. Configure OSPF for the ge-8/3/2.0 interface.

```
[edit]
user@router# set protocols ospf area 0.0.0.0 interface ge-8/3/2.0
```

27. Set up the static address for the physical interface module (PIM) rendezvous point (RP).

```
[edit]
user@router# set protocols pim rp static address 10.2.1.3
```

28. Configure the loopback interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface lo0.0 mode bidirectional-sparse
```

29. Set the ge-8/3/8.0 interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface ge-8/3/8.0 mode bidirectional-sparse
```

30. Configure the xe-0/1/3.0 interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface xe-0/1/3.0 mode bidirectional-sparse
```

31. Set the ge-8/3/2.0 interface to bidirectional sparse mode for the PIM protocol.

```
[edit]
user@router# set protocols pim interface ge-8/3/2.0 mode bidirectional-sparse
```

32. Configure redundant graceful switchover on the chassis.

```
[edit]
user@router# set chassis redundancy graceful-switchover
```

33. Set the aggregated ethernet device count to 10.

```
[edit]
user@router# set chassis aggregated-devices ethernet device-count 10
```

34. Configure the tunnel services bandwidth for FPC 1/PIC 0.

```
[edit]
user@router# set chassis fpc 1 pic 0 tunnel-services bandwidth 10g
```

35. Enable enhanced IP for network services on the chassis.

```
[edit]
user@router# set chassis network-services enhanced-ip
```

Results

From configuration mode, confirm your configuration by entering the following commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

user@router# show switch-options

```
switch-options {
  vtep-source-interface lo0.0;
}
```

user@router# show bridge-domains

```
bridge-domains {
  vlan-5 {
    vxlan {
      vni 100;
      multicast-group 239.1.1.1;
    }
    vlan-id 100;
    routing-interface irb.0;
    interface xe-1/0/0.0;
  }
```

```
    vlan-6 {
      vxlan {
        vni 200;
        multicast-group 239.2.1.1;
      }
      vlan-id 200;
      routing-interface irb.1;
      interface xe-2/0/0.0;
    }
  }
```

user@router# show interfaces

```
  interfaces {
    xe-1/0/0 {
      vlan-tagging;
      encapsulation flexible-ethernet-services;
      unit 0 {
        encapsulation vlan-bridge;
        vlan-id 100;
      }
    }
    xe-2/0/0 {
      vlan-tagging;
      encapsulation flexible-ethernet-services;
      unit 0 {
        encapsulation vlan-bridge;
        vlan-id 200;
      }
    }
    irb {
      unit 0 {
        family inet {
          address 5.5.5.1/24;
        }
      }
      unit 1 {
        family inet {
          address 6.6.6.1/24;
        }
      }
    }
    lo0 {
      unit 0 {
        family inet {
          address 3.3.3.3/32;
        }
      }
    }
  }
```

user@router# show protocols ospf

```
  area 0.0.0.0 {
    interface ge-8/3/8.0;
    interface lo0.0;
    interface xe-0/1/3.0;
    interface ge-8/3/2.0;
```

```
  }
user@router# show protocols pim

  rp {
    static {
      address 10.2.1.3;
    }
  }
user@router# show chassis

  redundancy {
    graceful-switchover;
  }
  aggregated-devices {
    ethernet {
      device-count 10;
    }
  }
  fpc 1 {
    pic 0 {
      tunnel-services {
        bandwidth 10g;
      }
    }
  }
  network-services enhanced-ip;
```

Verification

Confirm that the configuration is working properly.

- Verifying Reachability on page 52
- Verifying VXLAN on page 53

Verifying Reachability

Purpose Verify that the network is up and running with the proper interfaces and routes installed.

Action

```
user@router> show interfaces terse irb
Interface               Admin Link Proto    Local                 Remote
irb                     up    up
irb.0                   up    up   inet     5.5.5.1/24
                                  multiservice
irb.1                   up    up   inet     6.6.6.1/24
                                  multiservice
```

```
user@router> ping 5.5.5.1/24
PING 5.5.5.1 (5.5.5.1): 56 data bytes
64 bytes from 5.5.5.1: icmp_seq=0 ttl=64 time=0.965 ms
64 bytes from 5.5.5.1: icmp_seq=1 ttl=64 time=0.960 ms
64 bytes from 5.5.5.1: icmp_seq=2 ttl=64 time=0.940 ms
^C
--- 1.1.1.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.940/0.955/0.965/0.011 ms
```

Meaning Use the **show interfaces terse irb** command to verify that the IRB interface has been properly configured. The **irb.0** and **irb.1** interfaces should display the proper multiservice inet addresses.

Use the **ping** command to confirm that the network is connected to the IRB multiservice address.

Verifying VXLAN

Purpose Verify that VXLAN is working and the proper protocols are enabled.

Action

```
user@router> show interfaces vtep
Physical interface: vtep, Enabled, Physical link is Up
  Interface index: 133, SNMP ifIndex: 575
  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 334) (SNMP ifIndex 607)
    Flags: Up SNMP-Traps Encapsulation: ENET2
    VXLAN Endpoint Type: Source, VXLAN Endpoint Address: 10.255.187.32, L2 Routing
 Instance: default-switch, L3 Routing Instance: default
    Input packets : 0
    Output packets: 0
```

```
user@router> show l2-learning vxlan-tunnel-end-point remote 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)

Logical system   : <default>
Routing instance : default-switch
 Bridging domain : vlan-5+100, VLAN : 100, VNID : 100
 Bridging domain : vlan-6+200, VLAN : 200, VNID : 200
```

```
user@router> show l2-learning vxlan-tunnel-end-point source
Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
<default>                 0   10.255.187.32    lo0.0    0
    L2-RTT                   Bridge Domain              VNID     MC-Group-IP
    default-switch           vlan-5+100                 100      239.1.1.1

    default-switch           vlan-6+200                 200      239.1.1.1
```

Meaning

Use the **show interface vtep** command to displays information about VXLAN endpoint configuration. Make sure the routing instance is assigned to the default-switch..

Use the **show l2-learning vxlan-tunnel-end-point remote mac-table** command to confirm that the bridging domain VLAN groups were configured correctly.

Use the **show l2-learning vxlan-tunnel-end-point source** command to confirm the multicast IP addresses for bridging domain VLAN groups.

Related Documentation

- Understanding VXLANs on page 15
- show bridge mac-table on page 108
- show vpls mac-table on page 127

Example: Configuring VXLAN to VPLS Stitching with OVSDB

Supported Platforms EX9200, MX Series

Virtual Extensible LAN (VXLAN) can be utilized with the Open vSwitch Database (OVSDB) management protocol in a VPLS-enabled network to stitch a virtualized data center into a Layer 2 VPN network. This configuration allows for seamless interconnection between different data centers using Layer 2 VPN regardless of whether it is virtualized, physical, or both.

Requirements

This example uses the following hardware and software components:

- Two MX Series routers running Junos OS 14.1R2 or later
- Two MX Series routers running Junos OS 14.1R2 or later with an OVSDB software package. The release of this package must be the same as the Junos OS release running on the device.
- One EX9200 switch
- One VMware NSX controller
- NSX Manager

Before you start the configuration, you must perform the following tasks:

- In NSX Manager or the NSX API, configure a logical switch for each VXLAN that OVSDB will manage. This example implements two OVSDB-managed VXLANs, so you must configure two logical switches. After the configuration of each logical switch, NSX automatically generates a universally unique identifier (UUID) for the logical switch. If you have not done so already, retrieve the UUID for each logical switch. A sample UUID is 28805c1d-0122-495d-85df-19abd647d772. When configuring the equivalent VXLANs on the Juniper Networks device, you must use the UUID of the logical switch as the bridge domain name.

  For more information about logical switches and VXLANs, see "Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment" on page 10.

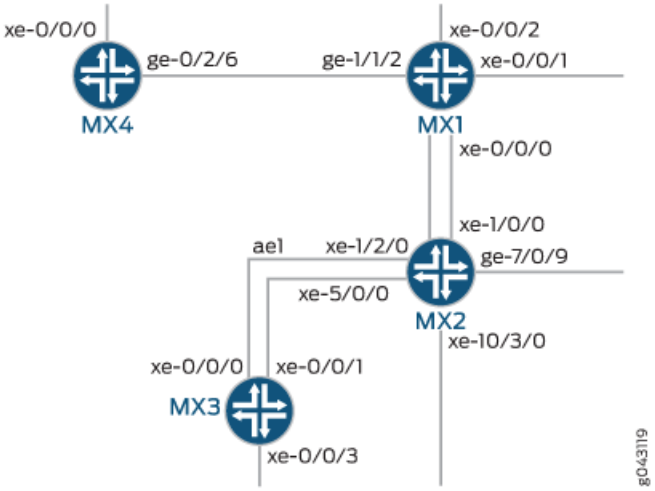
- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the Juniper Networks device. For more information, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers" on page 80.

Overview

In this example, four MX Series routers are configured to function together for VXLAN to virtual private LAN service (VPLS) stitching. Each router performs a different role in the configuration. The following diagram shows the topology of these MX Series routers. MX1 is the core router that handles Layer 3 traffic and protocols. MX2 is the VXLAN

gateway router that functions as a virtual tunnel endpoint (VTEP) and handles switching for Layer 2, VPLS, and VXLAN. The MX3 router is configured to handle VPLS traffic. The MX4 router is configured as a VTEP to accept and decapsulate VXLAN packets.

Topology



Configuration

To configure VXLAN to VPLS stitching with OVSDB:

- Configuring MX1 on page 59
- Configuring MX2 on page 59
- Configuring MX3 on page 62
- Configuring MX4 on page 63
- Results on page 64

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.

MX1

```
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
set interfaces lo0 unit 0 family inet address 10.255.181.13/32primary
set interfaces ge-1/1/2 unit 0 family inet address 30.30.30.2/30
set interfaces xe-0/0/0 unit 0 family inet address 20.20.20.1/30
set protocols ospf area 0.0.0.0 interface all
```

MX2

```
set interfaces lo0 unit 0 family inet address 10.255.181.72/32primary
set lag-options interfaces <ae*> mtu 9192
set lag-options interfaces <ae*> aggregated ether-options minimum-links 1
set chassis aggregated-devices ethernet device-count 40
set chassis fpc 1 pic 0 tunnel-services bandwidth 10g
set chassis network-services enhanced-ip
```

```
set interfaces xe-1/2/0 gigether-options 802.3ad ae1
set interfaces xe-0/0/0 mtu 1600
set interfaces xe-0/0/0 unit 0 family inet address 20.20.20.1/30
set interfaces ge-7/0/9 unit 1 vlan-id 3
set interfaces ge-7/0/9 unit 1 family vpls
set interfaces xe-10/3/0 vlan tagging
set interfaces xe-10/3/0 unit 0 vlan-id 2
set interfaces xe-10/3/0 unit 0 family vpls
set interfaces ae1 unit 0 family inet address 1.1.1.1/30
set interfaces ae1 unit 0 family mpls
set routing-options autonomous-system 100
set protocols rsvp interface all
set protocols mpls no cspf
set protocols mpls label-switched-path-to-nick to 10.255.181.98
set protocols mpls interface all
set protocols bgp family l2vpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.98 local-address 10.255.181.72
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
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 ae1.0
set protocols ovsdb traceoptions file ovsdb.log size 100m files 10
set protocols ovsdb traceoptions file ovsdb.level all
set protocols ovsdb traceoptions file ovsdb.flag all
set protocols ovsdb traceoptions file ovsdb.flag all
set protocols ovsdb interfaces xe-10/3/0.1
set protocols ovsdb interfaces xe-10/3/0.0
set protocols ovsdb interfaces ge-7/0/9.0
set protocols ovsdb interfaces ge-7/0/9.1
set protocols ovsdb controller 192.168.182.45 protocol ssl port 6632
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vtep-source-interface
  lo0.0
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 instance-type vpls
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 interface ge-7/0/9.1
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 interface xe-10/3/0.1
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 routing-interface irb.3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 route-distinguisher
  10.255.181.72:3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 vrf-target target:3:3
set routing-instances 24a76aff-7e61-4520-a78d-3eca26ad7510 protocols vpls site mx2
  site-identifier 1
set switch-options vtep-source-interface lo0.0
```

MX3

```
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
set interfaces lo0 unit 0 family inet address 10.255.181.98/32 primary
set lag-options interfaces <ae*> mtu 9192
set lag-options interfaces <ae*> aggregated-ether-options minimum-links 1
set lag-options interfaces <ae*> aggregated-ether-options lacp active
set interfaces xe-0/0/0 gigether-options 802.3ad ae1
set interfaces xe-0/0/1 gigether-options 802.3ad ae1
set interfaces xe-0/0/3 vlan tagging
set interfaces xe-0/0/3 unit 1 vlan-id 3
```

```
set interfaces xe-0/0/3 unit 1 family vpls
set interfaces ae1 mtu 9192
set interfaces ae1 unit 0 family inet address 1.1.1.2/30
set interfaces ae1 unit 0 family mpls
set routing-options autonomous-system 100
set protocols rsvp interface all
set protocols mpls no-cspf
set protocols mpls label-switched-path-to-mx2 to 10.255.181.72
set protocols mpls interface all
set protocols bgp family l2vpn signaling
set protocols bgp group ibgp type internal
set protocols bgp group ibgp neighbor 10.255.181.72 local-address 10.255.181.98
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ae1.0
set routing-instances vpls3 instance-type vpls
set routing-instances vpls3 vlan-id 3
set routing-instances interface xe-0/0/3.1
set routing-instances route-distinguisher 10.255.181.98:3
set routing-instances vrf-target target:3:3
set routing-instances protocols vpls no-tunnel-services
set routing-instances protocols vpls site nick site-identifier 2
```

MX3 old

```
set groups global interfaces lo0 unit 0 family inet address 127.0.0.1/32
set groups global interfaces lo0 unit 0 family inet address 10.255.181.98/32 primary
set groups global interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5518.1098.00
set groups global interfaces lo0 unit 0 family inet6 address abcd::10:255:181:98/128
  primary
set interfaces xe-0/0/0 gigether-options 802.3ad ae1
set interfaces xe-0/0/1 gigether-options 802.3ad ae1
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 0 family bridge interface-mode trunk
set interfaces xe-0/0/3 unit 0 family bridge vlan-id-list 1-40
set interfaces ae1 vlan-tagging
set interfaces ae1 encapsulation flexible-ethernet-services
set interfaces ae1 unit 0 family bridge interface-mode trunk vlan-id-list 1-40
set routing-instances vs1 instance-type virtual-switch
set routing-instances vs1 interface xe-0/0/3.0
set routing-instances vs1 interface ae1.0
set routing-instances vs1 bridge-domains v1 vlan-id-list 1-40
```

MX4

```
set interfaces lo0 unit 0 family inet address 10.255.181.43/32 primary
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 0 family bridge interface-mode trunk
set interfaces xe-0/0/0 unit 0 family bridge vlan-id-list 1-10
set interfaces ge-0/2/6 unit 0 family inet address 30.30.30.1/30
set protocols ospf area 0.0.0.0 interface ge-0/2/6.0
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 l2-learning traceoptions file vxlan-l2ald.log size 100m files 10
set protocols l2-learning traceoptions level all
set protocols l2-learning traceoptions flag all
set protocols layer2-control nonstop-bridging
```

```
set protocols ovsdb traceoptions file ovsdb.log size 100m files 10
set protocols ovsdb traceoptions level all
set protocols ovsdb traceoptions flag all
set protocols ovsdb interfaces xe-0/0/0.1
set protocols ovsdb interfaces xe-0/0/0.0
set protocols ovsdb controller 192.168.182.45 protocol ssl port 6632
set routing-instances default-vs1 vtep-source-interface lo0.0
set routing-instances default-vs1 instance-type virtual-switch
set routing-instances default-vs1 interface xe-0/0/0.1
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 3
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 3
set bridge-domains 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan
  ingress-node-replication
set switch-options vtep-source-interface lo0.0
```

Configuring MX1

Step-by-Step Procedure

The first router to be configured is the core router. This MX Series router handles Layer 3 traffic and protocols for the rest of the network.

To configure the MX1 router:

1. Specify the IPv4 address for the loopback interface.

```
[edit interfaces]
user@MX1# set lo0 unit 0 family inet address 127.0.0.1/32
user@MX1# set lo0 unit 0 family inet address 10.255.181.13/32primary
```

2. Configure the Layer 3 network.

```
[edit interfaces]
user@MX1# set ge-1/1/2 unit 0 family inet address 30.30.30.2/30
user@MX1# set xe-0/0/0 unit 0 family inet address 20.20.20.1/30
```

3. Enable OSPF on all interfaces.

```
[edit protocols]
user@MX1# set ospf area 0.0.0.0 interface all
```

Configuring MX2

Step-by-Step Procedure

The second router to be configured is the VXLAN gateway router. This MX Series router is configured as a VTEP, and it handles switching for Layer 2, VPLS, and VXLAN.

To configure the MX2 router:

1. Configure interfaces for the VXLAN gateway.

```
[edit interfaces]
user@MX2# set xe-1/2/0 gigether-options 802.3ad ae1
user@MX2# set xe-5/0/0 gigether-options 802.3ad ae1
user@MX2# set ge-7/0/0 gigether-options 802.3ad ae2
user@MX2# set ge-7/0/1 mtu 1600
user@MX2# set ge-7/0/1 unit 0 family inet address 20.20.20.1/30
user@MX2# set ge-7/1/3 gigether-options 802.3ad ae2
user@MX2# set xe-10/3/0 vlan-tagging
```

```
user@MX2# set xe-10/3/0 encapsulation flexible-ethernet-services
user@MX2# set xe-10/3/0 unit 100 family bridge interface-mode trunk
user@MX2# set xe-10/3/0 unit 100 family bridge vlan-id-list 100-101
user@MX2# set xe-10/3/0 unit 102 family bridge interface-mode trunk
user@MX2# set xe-10/3/0 unit 102 family bridge vlan-id-list 102-103
user@MX2# set ae1 vlan-tagging
user@MX2# set ae1 encapsulation flexible-ethernet-services
user@MX2# set ae1 unit 100 family bridge interface-mode trunk
user@MX2# set ae1 unit 100 family bridge vlan-id-list 100-101
user@MX2# set ae1 unit 102 family bridge interface-mode trunk
user@MX2# set ae1 unit 102 family bridge vlan-id-list 102-103
user@MX2# set ae2 mtu 1600
user@MX2# set ae2 unit 0 family inet address 20.20.20.5/30
user@MX2# set lo0 unit 1 family inet address 200.1.1.1/32
```

2. Configure OSPF interface settings and the Layer 2 learning traceoption file.

```
[edit protocols]
user@MX2# set ospf area 0.0.0.0 interface ge-7/0/1.0
user@MX2# set ospf area 0.0.0.0 interface fxp0.0 disable
user@MX2# set ospf area 0.0.0.0 interface lo0.0 passive
user@MX2# set l2-learning traceoptions file vxlan-l2ald.log
user@MX2# set l2-learning traceoptions file size 100m
user@MX2# set l2-learning traceoptions file files 10
user@MX2# set l2-learning traceoptions level all
user@MX2# set l2-learning traceoptions flag all
user@MX2# set layer2-control nonstop-bridging
```

3. Set up OVSDB tracing operations.

```
[edit protocols]
user@MX2# set ovsdb traceoptions file ovsdb.log
user@MX2# set ovsdb traceoptions file size 100m
user@MX2# set ovsdb traceoptions file files 10
user@MX2# set ovsdb traceoptions level all
user@MX2# set ovsdb traceoptions flag all
```

4. Specify that interfaces xe-10/3/0.1, xe-10/3/0.0, ae1.0, and ae1.1 are managed by OVSDB.

```
[edit protocols]
user@MX2# set ovsdb interfaces xe-10/3/0.1
user@MX2# set ovsdb interfaces xe-10/3/0.0
user@MX2# set ovsdb interfaces ae1.0
user@MX2# set ovsdb interfaces ae1.1
```

5. Configure a connection with an NSX controller.

```
[edit protocols]
user@MX2# set ovsdb controller 192.168.182.45 protocol ssl port 6632
```

6. Configure the **default-VS1** virtual switch instance as a VTEP.

```
[edit routing-instances]
user@MX2# set default-VS1 vtep-source-interface lo0.1
user@MX2# set default-VS1 instance-type virtual-switch
user@MX2# set default-VS1 interface xe-10/3/0.102
user@MX2# set default-VS1 interface ae1.102
```


7. Configure a set of bridge domains that are associated with VXLAN under the virtual switch instance.

```
[edit routing-instances]
user@MX2# set default-VS1 bridge-domains
  bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab vlan-id 102
user@MX2# set default-VS1 bridge-domains
  bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab vxlan ovsdb-managed
user@MX2# set default-VS1 bridge-domains
  bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab vxlan vni 102
user@MX2# set default-VS1 bridge-domains
  bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab vxlan ingress-node-replication
user@MX2# set default-VS1 bridge-domains
  f293dd5b-a901-4dba-bcbf-18a9979cf9d3 vlan-id 103
user@MX2# set default-VS1 bridge-domains
  f293dd5b-a901-4dba-bcbf-18a9979cf9d3 vxlan ovsdb-managed
user@MX2# set default-VS1 bridge-domains
  f293dd5b-a901-4dba-bcbf-18a9979cf9d3 vxlan vni 103
user@MX2# set default-VS1 bridge-domains
  f293dd5b-a901-4dba-bcbf-18a9979cf9d3 vxlan ingress-node-replication
```

8. Set up VPN routing and forwarding.

```
[edit routing- instances]
user@MX2# set vrf1 instance-type vrf
user@MX2# set vrf1 interface ae2.0
user@MX2# set vrf1 interface lo0.1
user@MX2# set vrf1 route-distinguisher 100:100
user@MX2# set vrf1 vrf-target target:100:100
user@MX2# set vrf1 protocols ospf area 0.0.0.0 interface ae2.0
user@MX2# set vrf1 protocols ospf area 0.0.0.0 interface lo0.1 passive
```

9. Configure bridge domains with VXLAN information.

```
[edit bridge-domains]
user@MX2# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 100
user@MX2# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
user@MX2# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 100
user@MX2# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan
  ingress-node-replication
user@MX2# set cadbc185-f60f-48a6-93fd-dc14a6420c60 vlan-id 101
user@MX2# set cadbc185-f60f-48a6-93fd-dc14a6420c60 vxlan ovsdb-managed
user@MX2# set cadbc185-f60f-48a6-93fd-dc14a6420c60 vxlan vni 101
user@MX2# set cadbc185-f60f-48a6-93fd-dc14a6420c60 vxlan
  ingress-node-replication
```

10. Configure the loopback interface to be used as the tunnel source address.

```
[edit switch-options]
user@MX2# set vtep-source-interface lo0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX equivalent of a hardware VTEP. This configuration implements one hardware VTEP, so you must configure one gateway, a gateway service, and a logical switch port using NSX Manager or the NSX API. For more information about the tasks you must perform as well as key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

Configuring MX3

Step-by-Step Procedure

The third MX Series router must be configured to handle VPLS traffic.

To configure the MX3 router:

1. Specify the IPv4, IPv6, and ISO addresses for the loopback interface.

```
[edit groups global interfaces]
user@MX3# set lo0 unit 0 family inet address 127.0.0.1/32
user@MX3# set lo0 unit 0 family inet address 10.255.181.98/32 primary
user@MX3# set lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5518.1098.00
user@MX3# set lo0 unit 0 family inet6 address abcd::10:255:181:98/128 primary
```

2. Configure the interfaces.

```
[edit interfaces]
user@MX3# set fxp0 unit 0 family inet address 192.168.181.97/25
user@MX3# set xe-0/0/0 gigether-options 802.3ad ae1
user@MX3# set xe-0/0/1 gigether-options 802.3ad ae1
user@MX3# set xe-0/0/3 vlan-tagging
user@MX3# set xe-0/0/3 encapsulation flexible-ethernet-services
user@MX3# set xe-0/0/3 unit 0 family bridge interface-mode trunk
user@MX3# set xe-0/0/3 unit 0 family bridge vlan-id-list 1-40
user@MX3# set ae1 vlan-tagging
user@MX3# set ae1 encapsulation flexible-ethernet-services
user@MX3# set ae1 unit 0 family bridge interface-mode trunk vlan-id-list 1-40
```

3. Configure the VPLS bridge domain and interfaces.

```
[edit routing-instances]
user@MX3# set vs1 instance-type virtual-switch
user@MX3# set vs1 interface xe-0/0/3.0
user@MX3# set vs1 interface ae1.0
user@MX3# set vs1 bridge-domains v1 vlan-id-list 1-40
```

Configuring MX4

Step-by-Step Procedure

The fourth MX Series router is configured as a VTEP to accept and decapsulate VXLAN packets.

To configure the MX4 router:

1. Specify the IPv4, IPv6, and ISO addresses for the loopback interface.

```
[edit interfaces]
user@MX4# set lo0 unit 0 family inet address 127.0.0.1/32
user@MX4# set lo0 unit 0 family inet address 10.255.181.43/32 primary
```

2. Configure the interfaces.

```
[edit interfaces]
user@MX4# set xe-0/0/0 vlan-tagging
user@MX4# set xe-0/0/0 encapsulation flexible-ethernet-services
user@MX4# set xe-0/0/0 unit 0 family bridge interface-mode trunk
user@MX4# set xe-0/0/0 unit 0 family bridge vlan-id-list 1-10
user@MX4# set ge-0/2/6 unit 0 family inet address 30.30.30.1/30
```

3. Configure OSPF interface settings and the Layer 2 learning traceoption file.

```
[edit protocols]
user@MX4# set ospf area 0.0.0.0 interface ge-0/2/6.0
user@MX4# set ospf area 0.0.0.0 interface fxp0.0 disable
user@MX4# set ospf area 0.0.0.0 interface lo0.0 passive
user@MX4# set l2-learning traceoptions file vxlan-l2ald.log size 100m files 10
user@MX4# set l2-learning traceoptions level all
user@MX4# set l2-learning traceoptions flag all
user@MX4# set layer2-control nonstop-bridging
```

4. Set up OVSDB tracing operations.

```
[edit protocols]
user@MX4# set ovsdb traceoptions file ovsdb.log size 100m files 10
user@MX4# set ovsdb traceoptions level all
user@MX4# set ovsdb traceoptions flag all
```

5. Specify that interfaces xe-0/0/0.1 and xe-0/0/0.0 are managed by OVSDB.

```
[edit protocols]
user@MX4# set ovsdb interfaces xe-0/0/0.1
user@MX4# set ovsdb interfaces xe-0/0/0.0
```

6. Configure a connection with an NSX controller.

```
[edit protocols]
user@MX4# set ovsdb controller 192.168.182.45 protocol ssl port 6632
```

7. Configure the VPLS interface.

```
[edit routing-instances]
user@MX4# set default-vs1 vtep-source-interface lo0.0
user@MX4# set default-vs1 instance-type virtual-switch
user@MX4# set default-vs1 interface xe-0/0/0.1
```

8. Configure a set of VXLAN-enabled bridge domains.

```
[edit bridge-domains]
```

```
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vlan-id 3
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan ovsdb-managed
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan vni 3
user@MX4# set 24a76aff-7e61-4520-a78d-3eca26ad7510 vxlan
  ingress-node-replication
```

9. Configure the loopback interface to be used as the tunnel source address.

```
[edit switch-options]
user@MX4# set vtep-source-interface lo0.0
```

NOTE: After completing this configuration, you must configure a gateway, which is the NSX equivalent of a hardware VTEP. This configuration implements one hardware VTEP, so you must configure one gateway, a gateway service, and a logical switch port using NSX Manager or the NSX API. For more information about the tasks you must perform as well as key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

Results

From configuration mode, confirm your configuration by entering the following commands on each router. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Verification

Confirm that the configuration is working properly.

Verifying MX1

Purpose Verify your configuration on MX1.

Action Verify that the interfaces are configured properly.

```
user@MX1# show interface

  apply-groups LAG-options;
    xe-0/0/2 {
      mtu 9000;
      unit 0 {
        family inet {
          address 80.80.0.250/24;
        }
      }
    }
    xe-0/0/3 {
      unit 0 {
        family inet {
          address 30.30.30.6/30;
        }
      }
    }
    ge-1/0/5 {
      mtu 1600;
      unit 0 {
        family inet {
          address 20.20.20.2/30;
        }
      }
    }
    ge-1/0/6 {
      unit 0 {
        family inet {
          address 20.20.20.10/30;
        }
      }
    }
    ge-1/0/7 {
      gigether-options {
        802.3ad ae2;
      }
    }
    ge-1/0/8 {
      gigether-options {
        802.3ad ae2;
      }
    }
    ge-1/1/2 {
      unit 0 {
        family inet {
          address 30.30.30.2/30;
        }
      }
    }
    ae2 {
      mtu 1600;
      unit 0 {
```

```
            family inet {
            address 20.20.20.6/30;
          }
        }
      }
```

Verify the loopback addresses.

```
user@MX1# show groups global interfaces
```

```
    lo0 {
      unit 0 {
        family inet {
          address 127.0.0.1/32;
          address 10.255.181.13/32 {
            primary;
          }
        }
        family iso {
          address 47.0005.80ff.f800.0000.0108.0001.0102.5518.1013.00;
        }
        family inet6 {
          address abcd::10:255:181:13/128 {
            primary;
          }
        }
      }
    }
```

Verify that OSPF is configured correctly.

```
user@MX1# show protocols
```

```
    ospf {
      area 0.0.0.0 {
        interface xe-0/0/2.0;
        interface ge-1/0/5.0;
        interface lo0.0 {
          passive;
        }
        interface fxp0.0 {
          disable;
        }
        interface ae2.0;
        interface ge-1/0/6.0;
        interface all;
      }
    }
```

Verifying MX2

Purpose Verify your configuration on MX2.

Action Verify that the interfaces are configured properly.

```
user@MX2# show interfaces
    xe-1/2/0 {
      gigether-options {
        802.3ad ae1;
      }
    }
    xe-5/0/0 {
      gigether-options {
        802.3ad ae1;
      }
    }
    ge-7/0/0 {
      gigether-options {
        802.3ad ae2;
      }
    }
    ge-7/0/1 {
      mtu 1600;
      unit 0 {
        family inet {
          address 20.20.20.1/30;
        }
      }
    }
    ge-7/1/3 {
      gigether-options {
        802.3ad ae2;
      }
    }
    xe-10/3/0 {
      vlan-tagging;
      encapsulation flexible-ethernet-services;
      unit 100 {
        family bridge {
          interface-mode trunk;
          vlan-id-list 100-101;
        }
      }
      unit 102 {
        family bridge {
          interface-mode trunk;
          vlan-id-list 102-103;
        }
      }
    }
    ae1 {
      vlan-tagging;
      encapsulation flexible-ethernet-services;
      unit 100 {
        family bridge {
          interface-mode trunk;
          vlan-id-list 100-101;
```

```
            }
        }
        unit 102 {
            family bridge {
                interface-mode trunk;
                vlan-id-list 102-103;
            }
        }
    }
    ae2 {
        mtu 1600;
        unit 0 {
            family inet {
                address 20.20.20.5/30;
            }
        }
    }
    lo0 {
        unit 1 {
            family inet {
                address 200.1.1.1/32;
            }
        }
    }
```

Verify that OSPF is configured properly.

user@MX2# show protocols ospf

```
    area 0.0.0.0 {
        interface ge-7/0/1.0;
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
    }
```

Verify that Layer 2 learning is configured properly.

user@MX2# show protocols l2-learning

```
l2-learning {
    traceoptions {
        file vxlan-l2ald.log size 100m files 10;
        level all;
        flag all;
    }
}
```

Verify that Layer 2 control is configured properly.

user@MX2# show protocols layer2-control

```
layer2-control {
    nonstop-bridging;
}
```

Verify that OVSDB is configured properly.

user@MX2# show protocols ovsdb

```
ovsdb {
    traceoptions {
        file ovsdb.log size 100m files 10;
        level all;
        flag all;
    }
    interfaces {
        xe-10/3/0.1;
        xe-10/3/0.0;
        ae1.0;
        ae1.1;
    }
    controller 192.168.182.45 {
        protocol {
            ssl port 6632;
        }
    }
}
```

Verify the **default-VS1** routing instance configuration.

user@MX2# show routing-instances

```
default-VS1 {
    vtep-source-interface lo0.0;
    instance-type virtual-switch;
    interface xe-10/3/0.102;
    interface ae1.102;
    bridge-domains {
        bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab {
            vlan-id 102;
            vxlan {
                ovsdb-managed;
                vni 102;
                ingress-node-replication;
            }
        }
        f293dd5b-a901-4dba-bcbf-18a9979cf9d3 {
            vlan-id 103;
            vxlan {
                ovsdb-managed;
                vni 103;
                ingress-node-replication;
            }
        }
    }
}
```

Verify the **vrf1** routing instance configuration.

user@MX2# show routing-instances

```
vrf1 {
    instance-type vrf;
```

```
    interface ae2.0;
  interface lo0.1;
  route-distinguisher 100:100;
  vrf-target target:100:100;
  protocols {
    ospf {
      area 0.0.0.0 {
        interface ae2.0;
        interface lo0.0 {
          passive;
        }
      }
    }
  }
```

Verify the bridge domains configuration.

```
user@MX2# show bridge-domains
```

```
  24a76aff-7e61-4520-a78d-3eca26ad7510 {
    vlan-id 100;
    vxlan {
      ovsdb-managed;
      vni 100;
      ingress-node-replication;
    }
  }
  cadbc185-f60f-48a6-93fd-dc14a6420c60 {
    vlan-id 101;
    vxlan {
      ovsdb-managed;
      vni 101;
      ingress-node-replication;
    }
  }
```

Verify that the loopback interface is used as the tunnel source address.

```
user@MX2# show switch-options
    vtep-source-interface lo0.0;
```

Verifying MX3

Purpose Verify your configuration on MX3.

Action Verify that the global group interfaces are configured properly.

user@MX3# show groups global interfaces

```
lo0 {
  unit 0 {
    family inet {
      address 127.0.0.1/32;
      address 10.255.181.98/32 {
        primary;
      }
    }
    family iso {
      address 47.0005.80ff.f800.0000.0108.0001.0102.5518.1098.00;
    }
    family inet6 {
      address abcd::10:255:181:98/128 {
        primary;
      }
    }
  }
}
```

Verify that the interfaces are configured properly.

user@MX3# show interfaces

```
xe-0/0/0 {
  gigether-options {
    802.3ad ae1;
  }
}
xe-0/0/1 {
  gigether-options {
    802.3ad ae1;
  }
}
xe-0/0/3 {
  vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 0 {
    family bridge {
      interface-mode trunk;
      vlan-id-list 1-40;
    }
  }
}
ae1 {
  vlan-tagging;
  encapsulation flexible-ethernet-services;
  unit 0 {
    family bridge {
      interface-mode trunk;
      vlan-id-list 1-40;
    }
  }
```

```
    }
```

Verify the VPLS bridge domain and interfaces configuration.

```
user@MX3# show routing-instances VS1
        instance-type virtual-switch;
        interface xe-0/0/3.0;
        interface ae1.0;
        bridge-domains {
            v1 {
                vlan-id-list 1-40;
            }
        }
```

Verifying MX4

Purpose Verify your configuration on MX4.

Action Verify that the global group interfaces are configured properly.

user@MX4# show groups global interfaces

```
lo0 {
  unit 0 {
    family inet {
      address 127.0.0.1/32;
      address 10.255.181.43/32 {
        primary;
      }
    }
    family iso {
      address 47.0005.80ff.f800.0000.0108.0001.0102.5518.1043.00;
    }
    family inet6 {
      address abcd::10:255:181:43/128 {
        primary;
      }
    }
  }
}
```

Verify that the interfaces are configured properly.

user@MX4# show interfaces

```
xe-0/0/0 {
    vlan-tagging;
    encapsulation flexible-ethernet-services;
    unit 0 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 1-10;
        }
    }
    unit 1 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 11-15;
        }
    }
    unit 2 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 21-30;
        }
    }
    unit 3 {
        family bridge {
            interface-mode trunk;
            vlan-id-list 31-40;
        }
    }
}
ge-0/2/6 {
    unit 0 {
        family inet {
            address 30.30.30.1/30;
        }
```

```
        }
    }
    ge-0/3/0 {
        unit 0 {
            family inet {
                address 3.3.3.2/30;
            }
        }
    }
```

Verify that the OSPF interface settings are configured properly.

```
user@MX4# show protocols ospf
        area 0.0.0.0 {
            interface ge-0/2/6.0;
            interface fxp0.0 {
                disable;
            }
            interface lo0.0 {
                passive;
            }
            interface ge-0/3/0.0;
        }
```

Verify that the Layer 2 learning traceoption file is configured properly.

```
user@MX4# show protocols l2-learning
        traceoptions {
            file vxlan-l2ald.log size 100m files 10;
            level all;
            flag all;
        }
```

Verify that Layer 2 control is configured properly.

```
user@MX4# show protocols layer2-control
        nonstop-bridging;
```

Verify that OVSDB is configured properly.

```
user@MX4# show protocols ovsdb
        traceoptions {
            file ovsdb.log size 100m files 10;
            level all;
            flag all;
        }
        interfaces {
            xe-0/0/0.1;
            xe-0/0/0.0;
        }
        controller 192.168.182.45 {
            protocol {
                ssl port 6632;
            }
        }
```

Verify the **default-VS1** routing instance configuration and bridge domains.

```
user@MX4# show routing-instances default-VS1
```

```
        vtep-source-interface lo0.0;
        instance-type virtual-switch;
        interface xe-0/0/0.1;
        bridge-domains {
            bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab {
                vlan-id 11;
                vxlan {
                    ovsdb-managed;
                    vni 16777214;
                    ingress-node-replication;
                }
            }
            f293dd5b-a901-4dba-bcbf-18a9979cf9d3 {
                vlan-id 12;
                vxlan {
                    ovsdb-managed;
                    vni 12;
                    ingress-node-replication;
                }
            }
            v13 {
                vlan-id 13;
                vxlan {
                    vni 13;
                    multicast-group 228.1.1.13;
                }
            }
            v14 {
                vlan-id 14;
                vxlan {
                    vni 14;
                    multicast-group 228.1.1.14;
                }
            }
            v15 {
                vlan-id 15;
                vxlan {
                    vni 15;
                    multicast-group 228.1.1.15;
                }
            }
        }
```

Verify the **default-VS2** routing instance configuration and bridge domains.

```
user@MX4# show routing-instances default-VS2
        bridge-domains {
            v21 {
                vlan-id 21;
                vxlan {
                    vni 21;
                    multicast-group 228.1.1.21;
                }
            }
            v22 {
                vlan-id 22;
                vxlan {
                    vni 22;
                    multicast-group 228.1.1.22;
                }
            }
```

```
            v23 {
                vlan-id 23;
                vxlan {
                    vni 23;
                    multicast-group 228.1.1.23;
                }
            }
            v24 {
                vlan-id 24;
                vxlan {
                    vni 24;
                    multicast-group 228.1.1.24;
                }
            }
            v25 {
                vlan-id 25;
                vxlan {
                    vni 25;
                    multicast-group 228.1.1.25;
                }
            }
        }
```

Verify that the bridge domains are configured properly.

```
user@MX4# show bridge-domains
    24a76aff-7e61-4520-a78d-3eca26ad7510 {
        vlan-id 3;
        vxlan {
            ovsdb-managed;
            vni 3;
            ingress-node-replication;
        }
    }
    cadbc185-f60f-48a6-93fd-dc14a6420c60 {
        vlan-id 2;
        vxlan {
            ovsdb-managed;
            vni 2;
            ingress-node-replication;
        }
    }
    v1 {
        vlan-id 1;
        vxlan {
            vni 1;
            multicast-group 228.1.1.1;
        }
    }
    v4 {
        vlan-id 4;
        vxlan {
            vni 4;
            multicast-group 228.1.1.4;
        }
    }
    v5 {
        vlan-id 5;
        vxlan {
            vni 5;
            multicast-group 228.1.1.5;
```

```
        }
    }
```

Verify that the loopback interface is used as the tunnel source address.

```
user@MX4# show switch-options
    vtep-source-interface lo0.0;
```

Related Documentation

- Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment on page 10
- Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers on page 80

CHAPTER 4

Configuration Tasks

Installing Open vSwitch Database Components on Juniper Networks Devices

Supported Platforms EX Series, MX Series, QFX Series standalone switches

To install Open vSwitch Database (OVSDB) components on a Juniper Networks device, you must copy the OVSDB software package to the Juniper Networks device and then install the package. The OVSDB software package name uses the following format:

jsdn-*packageID-release*

where:

- *packageID* identifies the package that must run on each Juniper Networks device.
- *release* identifies the OVSDB release; for example, 14.2. The OVSDB software release and the Junos OS release running on the device must be the same.

For information about OVSDB support on Juniper Networks devices and the software package for each device, see "Open vSwitch Database Support on Juniper Networks Devices" on page 3.

To install the OVSDB software package on a Juniper Networks device:

1. Download the software package to the Juniper Networks device.
2. If an OVSDB software package already exists on the Juniper Networks device, remove the package by issuing the **request system software delete** operational mode command.

   user@device> request system software delete *existing-ovsdb-package*

3. Install the new software package by using the **request system software add** operational mode command.

   ```
   user@device> request system software add path-to-ovsdb-package
   ```

Related Documentation

- Understanding the Open vSwitch Database Management Protocol Running on Juniper Networks Devices on page 6

Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers

Supported Platforms EX Series, MX Series, QFX Series standalone switches

To secure a connection between a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol and one or more VMware NSX controllers, the following Secure Sockets Layer (SSL) files must be present in the **/var/db/certs** directory on the device:

- vtep-privkey.pem
- vtep-cert.pem
- ca-cert.pem

You must create the vtep-privkey.pem and vtep-cert.pem files for the device, and then install the two files in the **/var/db/certs** directory on the device.

Upon the initial connection between a Juniper Networks device with OVSDB implemented and an NSX controller, the ca-cert.pem file is automatically generated, and then installed in the **/var/db/certs** directory on the device.

The procedure provided in this topic uses the OpenFlow public key infrastructure (PKI) management utility ovs-pki on a Linux computer to initialize a public key infrastructure (PKI) and create the **vtep-privkey.pem** and **vtep-cert.pem** files. (If you have an existing PKI on your Linux computer, you can skip the step to initialize a new one.) By default, the utility initializes the PKI and places these files in the **/usr/local/share/openvswitch/pki** directory of the Linux computer.

To create and install an SSL key and certificate on a Juniper Networks device:

1. Initialize a PKI if one does not already exist on your Linux computer.

   ```
   # ovs-pki init
   ```

2. On the same Linux computer on which the PKI exists, create a new key and certificate for the Juniper Networks device.

   ```
   # ovs-pki req+sign vtep
   ```

3. Copy only the **vtep-privkey.pem** and **vtep-cert.pem** files from the Linux computer to the **/var/db/certs** directory on the Juniper Networks device.

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81

Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices

Supported Platforms EX Series, MX Series, QFX Series standalone switches

To implement the Open vSwitch Database (OVSDB) management protocol on a Juniper Networks device, you must explicitly configure a connection to at least one VMware NSX controller, using the Junos OS CLI.

All NSX controller connections are made on the management interface (fxp0 or me0) of the Juniper Networks device. This connection is secured by using the Secure Sockets Layer (SSL) protocol. The default port number over which the connection is made is 6632.

You must also specify that any interface with a physical server is managed by OVSDB. By performing this configuration, you are essentially disabling the Juniper Networks device from learning about other Juniper Networks devices that function as hardware virtual tunnel endpoints (VTEPs) and the MAC addresses learned by the hardware VTEPs, and enabling OVSDB to learn about these elements.

Before setting up OVSDB on a Juniper Networks device, you must do the following:

- Ensure that the Juniper Networks device has an OVSDB software package installed, and that the OVSDB software package release is the same as the Junos OS release running on the device.
- Determine the IP address of the NSX controller.
- Create an SSL private key and certificate, and install them in the **/var/db/certs** directory of the Juniper Networks device. For more information, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers" on page 80.

To set up OVSDB on a Juniper Networks device:

1. Specify the IP address of the NSX controller.

   ```
   [edit protocols ovsdb]
   user@host# set controller ip-address
   ```
2. Specify SSL as the protocol that secures the connection.

   ```
   [edit protocols ovsdb controller ip-address]
   user@host# set protocol ssl
   ```
3. Set the number of the port over which the connection to the NSX controller is made.

   ```
   [edit protocols ovsdb controller ip-address protocol ssl]
   user@host# set port number
   ```
4. (Optional) Specify (in milliseconds) how long the connection can be inactive before an inactivity probe is sent.

   ```
   [edit protocols ovsdb controller ip-address]
   ```

```
user@host# set inactivity-probe-duration milliseconds
```

5. (Optional) Specify (in milliseconds) how long the device must wait before it can try to connect to the NSX controller again if the previous attempt failed.

```
[edit protocols ovsdb controller ip-address]
user@host# set maximum-backoff-duration milliseconds
```

6. (Optional) Repeat steps 1 through 5 to explicitly configure a connection to an additional NSX controller in the same cluster.

7. Specify the interfaces that you want OVSDB to manage.

```
[edit protocols ovsdb]
user@host# set interfaces interface-name unit logical-unit-number
```

NOTE: After completing this procedure, you must set up OVSDB-managed VXLANs. For more information, see "Configuring OVSDB-Managed VXLANs" on page 84.

Related Documentation

- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints

Supported Platforms EX Series, MX Series, QFX Series standalone switches

For each Juniper Networks Junos operating system (Junos OS) network device that you plan to deploy as a hardware virtual tunnel endpoint (VTEP) in a physical network, you must create a VMware NSX-equivalent entity, which is known as a *gateway*, in NSX Manager or in the NSX API. You must also map the gateway to a logical switch, which is the NSX equivalent of an Open vSwitch Database (OVSDB)-managed Virtual Extensible LAN (VXLAN) in the physical network. Performing this configuration enables connectivity between physical servers in the physical network and virtual machines (VMs) in the virtual network.

This topic provides a high-level summary of the tasks that you must perform to create a gateway. Although you can create a gateway either in NSX Manager or in the NSX API, this topic describes the necessary tasks from the perspective of NSX Manager. Also, this topic does not include a complete procedure for each task. Rather, it includes key NSX Manager configuration details for ensuring the correct configuration of the virtual entities so that they function properly with the physical entities. For complete information about performing the tasks described in this topic, see the documentation that accompanies NSX Manager.

For each hardware VTEP that you deploy in the physical network, you must perform the following tasks:

- Creating a Gateway on page 83
- Creating a Gateway Service on page 83
- Creating a Logical Switch Port on page 84

Creating a Gateway

In NSX Manager, you must create a gateway for each hardware VTEP that you implement in the physical network. Table 9 on page 83 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a gateway.

Before you begin this task, you must configure a logical switch in NSX Manager or in the NSX API for each OVSDB-managed VXLAN that you plan to implement in the physical network. For information about configuring a logical switch, see the documentation that accompanies NSX Manager or the NSX API.

Table 9: Create a Gateway in NSX Manager: Key Configurations

NSX Manager Configuration Page/Dialog Box	NSX Manager Configuration Field	How to Configure
Type	Transport Node Type	Select **Gateway**.
Properties	VTEP Enabled	Select **VTEP Enabled**.
Credential	Type	Select **Management Address**.
Credential	Management Address	Specify the management IP address of the Juniper Networks device.
Connections/Create Transport Connector	Transport Type	Select **VXLAN**.
Connections/Create Transport Connector	Transport Zone UUID	Select the UUID of an existing transport zone, or create a new transport zone.
Connections/Create Transport Connector	IP Address	Specify the IP address of the loopback interface (lo0) of the Juniper Networks device.

Creating a Gateway Service

In NSX Manager, you must create a gateway service for each hardware VTEP that you implement in the physical network. Table 10 on page 83 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a gateway service.

Before you start this task, make sure that you have configured the OVSDB-managed interfaces on the hardware VTEP. For information about configuring OVSDB-managed interfaces, see "Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices" on page 81.

Table 10: Create a Gateway Service in NSX Manager: Key Configurations

NSX Manager Configuration Page/Dialog Box	NSX Manager Configuration Field	How to Configure
Type	Gateway Service Type	Select **VTEP L2 Gateway Service**.

Table 10: Create a Gateway Service in NSX Manager: Key Configurations *(continued)*

NSX Manager Configuration Page/Dialog Box	NSX Manager Configuration Field	How to Configure
Transport Nodes/Edit Gateway	Transport Node	Select the gateway that you created for the hardware VTEP.
Transport Nodes/Edit Gateway	Port ID	Select an OVSDB-managed interface configured on the hardware VTEP.

Creating a Logical Switch Port

In NSX Manager, you must create a logical switch port for each hardware VTEP that you implement in the physical network. Table 11 on page 84 provides a summary of key configuration fields in NSX Manager and how to configure them when creating a logical switch port.

Before you start this task, you must configure a logical switch in NSX Manager or in the NSX API for each OVSDB-managed VXLAN that you plan to implement in the physical network. For information about configuring a logical switch, see the documentation that accompanies NSX Manager or the NSX API.

Table 11: Create a Logical Switch Port in NSX Manager: Key Configurations

NSX Manager Configuration Page/Dialog Box	NSX Manager Configuration Field	How to Configure
Logical Switch	Logical Switch UUID	Select the UUID of the logical switch that corresponds to the hardware VTEP.
Attachment	Attachment Type	Select **VTEP L2 Gateway**.
Attachment	VTEP L2 Gateway Service UUID	Select the UUID of the gateway service you created for the hardware VTEP.

Related Documentation

- Configuring OVSDB-Managed VXLANs on page 84

Configuring OVSDB-Managed VXLANs

Supported Platforms EX Series, MX Series

To implement the OVSDB management protocol on a Juniper Networks device, you must configure OVSDB-managed VXLANs.

For Layer 2 broadcast, unknown unicast, and multicast (BUM) traffic that originates in an OVSDB-managed VXLAN and is forwarded to interfaces within the same VXLAN, you can optionally enable ingress node replication. With this feature enabled, the Juniper Networks device handles the replication of these packets and the forwarding of the replicas to interfaces within the same OVSDB-managed VXLAN. For more information about using ingress node replication or a service node, which is the default way to handle Layer 2 BUM traffic, see "Understanding How Layer 2 BUM Traffic and Layer 3 Routed Multicast Traffic Are Handled in VXLANs Managed by OVSDB" on page 8.

NOTE: When Juniper Networks devices replicate Layer 2 BUM packets to a large number of remote software virtual tunnel endpoints (VTEPs), the performance of the Juniper Networks devices might be impacted.

Before you configure VXLANs on a Juniper Networks device, using the Junos OS CLI:

- For each OVSDB-managed VXLAN that you plan to configure on a Juniper Networks device, you must configure a logical switch in VMware NSX Manager or the NSX API. (For information about configuring a logical switch, see the documentation that accompanies NSX Manager or the NSX API.) Based on the name and VXLAN network identifier (VNI) that you configure for the logical switch, NSX automatically generates a universally unique identifier (UUID) for the logical switch. You must retain the UUID of the logical switch for use when configuring a corresponding VXLAN on the Juniper Networks device as described in the following procedure.
- You must perform the configuration described in "Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices" on page 81.

To configure an OVSDB-managed VXLAN on a Juniper Networks device:

1. Configure the VXLANs that you want OVSDB to manage. You can configure the VXLANs in the context of a bridge domain, VLAN, routing instance, or switching instance.

   NOTE: For the name of the bridge domain or VLAN, you must specify the UUID for the logical switch configured in NSX Manager or the NSX API.

   Bridge domains:

   ```
   [edit bridge-domains bridge-domain-name vxlan]
   user@host# set ovsdb-managed
   ```

   VLANs:

   ```
   [edit vlans vlan-name vxlan]
   user@device# set ovsdb-managed
   ```

   Bridge domains within the specified routing instance:

   ```
   [edit routing-instances routing-instance-name bridge-domains bridge-domain-name
   vxlan]
   ```

```
user@host# set ovsdb-managed
```

VLANs within the specified routing instance:

```
[edit routing-instances routing-instance-name vlans vlan-name vxlan]
user@device# set ovsdb-managed
```

Default switching instance within the specified routing instance:

```
[edit routing-instances routing-instance-name switch-options]
user@host# set ovsdb-managed
```

All VXLAN entities within the specified routing instance:

```
[edit routing-instances routing-instance-name vxlan]
user@host# set ovsdb-managed
```

2. (Optional) Enable ingress node replication to handle Layer 2 BUM traffic on interfaces in the same VXLAN in which the traffic originated. You can configure ingress node replication in the context of a bridge domain, VLAN, or routing instance.

   Bridge domains:

   ```
   [edit bridge-domains bridge-domain-name vxlan]
   user@host# set ingress-node-replication
   ```

   VLANs:

   ```
   [edit vlans vlan-name vxlan]
   user@device# set ingress-node-replication
   ```

   Bridge domains, VLANs, or all VXLAN entities, respectively, within the specified routing instance:

   ```
   [edit routing-instances routing-instance-name bridge-domains bridge-domain-name
   vxlan]
   [edit routing-instances routing-instance-name vlans vlan-name vxlan]
   [edit routing-instances routing-instance-name vxlan]
   user@host# set ingress-node-replication
   ```

3. For each Juniper Networks device that you plan to implement as a hardware VTEP, you must perform some configuration tasks in NSX Manager or in the NSX API.

   For more information about the tasks you must perform and key NSX Manager configuration details, see "VMware NSX Configuration for Juniper Networks Devices That Function as Virtual Tunnel Endpoints" on page 82.

Related Documentation

- Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment on page 10
- Example: Setting Up Inter-VXLAN Unicast Routing and OVSDB Connections in a Data Center on page 21
- Example: Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center on page 32

CHAPTER 5

OVSDB Configuration Statements

controller (OVSDB)

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax

```
controller ip-address {
  inactivity-probe-duration milliseconds;
  maximum-backoff-duration milliseconds;
  protocol protocol {
    port number;
  }
}
```

Hierarchy Level [edit protocols ovsdb]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure a connection between a Juniper Networks device and a VMware NSX controller. The Junos OS device must be running a release that supports the Open vSwitch Database (OVSDB) management protocol and have the OVSDB software package installed. The OVSDB software package release must be the same as the Junos OS release running on the device.

The Junos OS implementation of OVSDB supports one cluster of NSX controllers, which includes three or five controllers according to VMware recommendations.

To implement OVSDB on a Junos OS device, you must explicitly configure a connection to at least one NSX controller, using the Junos OS CLI. If the NSX controller to which you explicitly configure a connection is in a cluster, the controller pushes information about other controllers in the same cluster to the device, and the device establishes connections with the other controllers. However, you can also explicitly configure connections with the other controllers in the cluster, using the Junos OS CLI.

Options ***ip-address***—IPv4 address of the NSX controller.

The remaining statements are explained separately.

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

inactivity-probe-duration

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax inactivity-probe-duration *milliseconds*;

Hierarchy Level [edit protocols ovsdb controller]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure the maximum amount of time, in milliseconds, that the connection between a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol and a VMware NSX controller can be inactive before an inactivity probe is sent.

Options ***milliseconds***—Number of milliseconds that the connection can be inactive before an inactivity probe is sent.

Range: 0 through 4,294,967,295

Default: 0. This value indicates that an inactivity probe is never sent.

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

ingress-node-replication

Supported Platforms EX Series, MX Series

Syntax ingress-node-replication;

Hierarchy Level [edit bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit routing-instances *routing-instance-name* vxlan],
[edit vlans *vlan-name* vxlan]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Enable ingress node replication for a specified Virtual Extensible LAN (VXLAN) that is managed by the Open vSwitch Database (OVSDB) management protocol.

With this feature enabled, instead of service nodes, Juniper Networks devices with OVSDB implemented handle incoming broadcast, unknown unicast, or multicast (BUM) traffic. For more information about the scenarios in which you can use ingress node replication and how it works, see "Understanding How Layer 2 BUM Traffic and Layer 3 Routed Multicast Traffic Are Handled in VXLANs Managed by OVSDB" on page 8.

NOTE: When Juniper Networks devices replicate Layer 2 BUM packets to a large number of remote software VTEPs, the performance of the Juniper Networks devices might be impacted.

Default If you do not include the **ingress-node-replication** statement, one or more service nodes handle BUM traffic.

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Configuring OVSDB-Managed VXLANs on page 84

interfaces (OVSDB)

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax interfaces *interface-name*;

Hierarchy Level [edit protocols ovsdb]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Specify the interfaces to be managed by the Open vSwitch Database (OVSDB) management protocol. Typically, the only interfaces that need to be managed by OVSDB are interfaces with physical servers.

Options *interface-name*—Name of the interface, including the logical unit number—for example, xe-1/1/0.0.

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 84

maximum-backoff-duration

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax maximum-backoff-duration *milliseconds*;

Hierarchy Level [edit protocols ovsdb controller]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Specify (in milliseconds) how long a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol waits before it re-attempts to connect with a VMware NSX controller if a previous attempt failed.

Options *milliseconds*—Number of milliseconds a Juniper Networks device waits before it re-attempts to connect with an NSX controller.

Range: 1000 through 4,294,967,295

Default: 1000

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

ovsdb

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax

```
ovsdb {
  controller ip-address {
    inactivity-probe-duration milliseconds;
    maximum-backoff-duration milliseconds;
    protocol protocol {
      port number;
    }
  }
  interfaces interface-name;
  traceoptions {
    file <filename> <files number> <match regular-expression> <no-world-readable |
      world-readable> <size size>;
    flag flag;
    no-remote-trace;
  }
}
```

Hierarchy Level [edit protocols]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure support for the Open vSwitch Database (OVSDB) management protocol on a Juniper Networks device. The Juniper Networks device must be running a release that supports OVSDB and have the OVSDB software package installed. The OVSDB software package release must be the same as the Junos OS release that is running on the device.

The remaining statements are explained separately.

Default The OVSDB management protocol is disabled on Juniper Networks devices.

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Understanding the Open vSwitch Database Management Protocol Running on Juniper Networks Devices on page 6
- Configuring OVSDB-Managed VXLANs on page 84

ovsdb-managed

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax ovsdb-managed;

Hierarchy Level
[edit bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* switch-options],
[edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit routing-instances *routing-instance-name* vxlan],
[edit switch-options],
[edit vlans *vlan-name* vxlan]

Release Information
Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Disable a Junos OS device from learning about other Junos OS devices that function as hardware virtual tunnel endpoints (VTEPs) in a specified Virtual Extensible LAN (VXLAN) and the MAC addresses learned by the hardware VTEPs. Instead, the Junos OS device uses the Open vSwitch Database (OVSDB) management protocol to learn about the hardware VTEPs in the VXLAN and the MAC addresses learned by the hardware VTEPs.

The specified VXLAN must have a VXLAN Network Identifier (VNI) configured, using the **vni** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instance *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy.

Also, this implementation of OVSDB uses the multicast scheme described in "Understanding How Layer 2 BUM Traffic and Layer 3 Routed Multicast Traffic Are Handled in VXLANs Managed by OVSDB" on page 8. Therefore, specifying the **multicast-group** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instances *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy has no effect.

Required Privilege Level
admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 84

port (OVSDB)

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax port *number*;

Hierarchy Level [edit protocols ovsdb controller protocol]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Specify the VMware NSX controller port to which a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol connects.

Options ***number***—Port number of NSX controller port.

Range: 1024 through 65,535

Default: 6632

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

protocol (OVSDB)

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax

```
protocol protocol {
  port number;
}
```

Hierarchy Level [edit protocols ovsdb controller]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Configure the security protocol that protects the connection between a Juniper Networks device that supports the Open vSwitch Database (OVSDB) management protocol and a VMware NSX controller.

The Secure Sockets Layer (SSL) connection requires a private key and certificates, which must be stored in the **/var/db/certs** directory of the Juniper Networks device. For more information about the files, including actions you must take to create and install some of the files, see "Creating and Installing an SSL Key and Certificate on a Juniper Networks Device for Connection with VMware NSX Controllers" on page 80.

Options ***protocol***—Establish a secure connection to the NSX controller, using SSL or the Transmission Control Protocol (TCP).

NOTE: **SSL is the only supported connection protocol.**

Default: ssl

The remaining statement is explained separately.

Required Privilege Level admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

traceoptions (OVSDB)

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax

```
traceoptions {
  file <filename> <files number> <match regular-expression> <no-world-readable |
    world-readable> <size size>;
  flag flag;
  no-remote-trace;
}
```

Hierarchy Level [edit protocols ovsdb]

Release Information Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Define tracing operations for the Open vSwitch Database (OVSDB) management protocol, which is supported on Juniper Networks devices.

Default If you do not include this statement, OVSDB-specific tracing operations are not performed.

Options

file *filename*—Name of file in which the system places the output of the tracing operations. By default, the system places all files in the **/var/log** directory.

Default: **/var/log/vgd**

files *number*—(Optional) Maximum number of trace files. When a trace file reaches the size specified by the **size** option, the filename is appended with 0 and compressed. For example, a trace file named **trace-file.gz** would be renamed **trace-file.0.gz**. When **trace-file.0.gz** reaches the specified size, it is renamed **trace-file.1.gz** and its contents are compressed to **trace-file.0.gz**. 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 and a filename.

Range: 2 through 1000 files

Default: 10 files

flag *flag*—Tracing operation to perform. You can include one or more of the following flags:

all—All OVSDB events.

configuration—OVSDB configuration events.

core—OVSDB core events.

function—OVSDB function events.

interface—OVSDB interface events.

l2-client—OVSDB Layer 2 client events.

ovs-client—OVSDB client events.

match *regular-expression*—(Optional) Only log lines that match the regular expression.

no-remote-trace—(Optional) Disable tracing and logging operations that track normal operations, error conditions, and packets that are generated by or passed through the Juniper Networks device.

no-world-readable—Restrict access to the trace files to the owner.

Default: no-world-readable

size *size*—(Optional) Maximum size of each trace file in bytes, kilobytes (KB), megabytes (MB), or gigabytes (GB). If you do not specify a unit, the default is bytes. If you specify a maximum file size, you also must specify a maximum number of trace files by using the **files** option and a filename by using the **file** option.

Syntax: *size* to specify bytes, *size*k to specify KB, *size*m to specify MB, or *size*g to specify GB.

Range: 10,240 through 1,073,741,824 bytes

Default: 128 KB

world-readable—Enable any user to access the trace files.

Required Privilege Level

admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation

- Example: Setting Up Inter-VXLAN Unicast Routing and OVSDB Connections in a Data Center on page 21
- Example: Setting Up Inter-VXLAN Unicast and Multicast Routing and OVSDB Connections in a Data Center on page 32

CHAPTER 6

VXLAN Configuration Statements

decapsulate-accept-inner-vlan

Supported Platforms QFX Series standalone switches

Syntax decapsulate-accept-inner-vlan

Hierarchy Level [edit protocols l2-learning]

Release Information Statement modified in Junos OS 14.1X53 for the QFX Series.

Description Configure the switch to de-encapsulate and accept original VLAN tags in VXLAN packets.

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 15
- **encapsulate-inner-vlan on page 100**

encapsulate-inner-vlan

Supported Platforms QFX Series standalone switches

Syntax encapsulate-inner-vlan

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Configure the switch to preserve the original VLAN tag (in the inner Ethernet packet) when performing 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 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*
- **decapsulate-accept-inner-vlan on page 99**

multicast-group

Supported Platforms QFX Series standalone switches

Syntax multicast-group

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Assign a multicast group address to a VXLAN. All members of a VXLAN must use the same multicast group address

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 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

ovsdb-managed

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax ovsdb-managed;

Hierarchy Level
[edit bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* bridge-domains *bridge-domain-name* vxlan],
[edit routing-instances *routing-instance-name* switch-options],
[edit routing-instances *routing-instance-name* vlans *vlan-name* vxlan],
[edit routing-instances *routing-instance-name* vxlan],
[edit switch-options],
[edit vlans *vlan-name* vxlan]

Release Information
Statement introduced in Junos OS Release 14.1R2.
Statement introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Statement introduced in Junos OS Release 14.2 for EX Series switches.

Description Disable a Junos OS device from learning about other Junos OS devices that function as hardware virtual tunnel endpoints (VTEPs) in a specified Virtual Extensible LAN (VXLAN) and the MAC addresses learned by the hardware VTEPs. Instead, the Junos OS device uses the Open vSwitch Database (OVSDB) management protocol to learn about the hardware VTEPs in the VXLAN and the MAC addresses learned by the hardware VTEPs.

The specified VXLAN must have a VXLAN Network Identifier (VNI) configured, using the **vni** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instance *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy.

Also, this implementation of OVSDB uses the multicast scheme described in "Understanding How Layer 2 BUM Traffic and Layer 3 Routed Multicast Traffic Are Handled in VXLANs Managed by OVSDB" on page 8. Therefore, specifying the **multicast-group** statement in the **[edit bridge-domains *bridge-domain-name* vxlan]**, **[edit routing-instances *routing-instance-name* vxlan]**, or **[edit vlans *vlan-name* vxlan]** hierarchy has no effect.

Required Privilege Level
admin—To view this statement in the configuration.
admin-control—To add this statement to the configuration.

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 84

unreachable-vtep-aging-timer

Supported Platforms QFX Series standalone switches

Syntax unreachable-vtep-aging-timer [300–1800]

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Configure the system to age out the address for the remote VTEP if all the MAC addresses learned from that VTEP age out. The address for the remote VTEP expires the configured number of seconds after the last learned MAC address expires.

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 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

vni

Supported Platforms QFX Series standalone switches

Syntax vni [1–16777214]

Hierarchy Level [edit vlans *VLAN* vxlan]

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Assign a numeric value to identify a VXLAN. All members of a VXLAN must use the same VNI.

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 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

vtep-source-interface

Supported Platforms QFX Series standalone switches

Syntax vtep-source-interface *logical-interface*;

Hierarchy Level [edit switch-options,

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description Configure a source interface for a VXLAN tunnel. You must provide the name of a logical interface configured on the 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 VXLANs on page 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

vxlan

Supported Platforms QFX Series standalone switches

Syntax

```
vxlan {
  encapsulate-inner-vlan
  multicast-group
  ovsdb-managed
  unreachable-vtep-aging-timer
  vni
  }
}
```

Hierarchy Level [edit vlans],

Release Information Statement introduced in Junos OS Release 14.1X53-D10.

Description

Options 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

- Understanding VXLANs on page 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

PART 3

Administration

CHAPTER 7

OVSDB Monitoring Commands

- show bridge mac-table
- show ovsdb controller
- show ovsdb interface
- show ovsdb logical-switch
- show ovsdb mac
- show ovsdb statistics interface
- show ovsdb virtual-tunnel-end-point
- show vpls mac-table

show bridge mac-table

Supported Platforms MX Series

Syntax
show bridge mac-table
<brief | count | detail | extensive>
<bridge-domain (all | *bridge-domain-name*)>
<global-count>
<interface *interface-name*>
<*mac-address*>
<vlan-id (all-vlan | *vlan-id*)>

Release Information Command introduced in Junos OS Release 8.4.

Description (MX Series routers only) Display Layer 2 MAC address information.

Options
none—Display all learned Layer 2 MAC address information.

brief | count | detail | extensive—(Optional) Display the specified level of output.

bridge-domain (all | *bridge-domain-name*)—(Optional) Display learned Layer 2 MAC addresses for all bridging domains or for the specified bridging domain.

global-count—(Optional) Display the total number of learned Layer 2 MAC addresses on the system.

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.

mac-address—(Optional) Display the specified learned Layer 2 MAC address information.

vlan-id (all-vlan | *vlan-id*)—(Optional) Display learned Layer 2 MAC addresses for all VLANs or for the specified VLAN.

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

List of Sample Output
show bridge mac-table on page 109
show bridge mac-table (with VXLAN enabled) on page 110
show bridge mac-table count on page 110
show bridge mac-table detail on page 111

Output Fields Table 12 on page 109 describes the output fields for the **show bridge mac-table** command. Output fields are listed in the approximate order in which they appear.

Table 12: show bridge mac-table Output fields

Field Name	Field Description
Routing instance	Name of the routing instance.
Bridging domain	Name of the bridging domain.
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. • **C**—Control MAC address is configured. • **SE**—MAC accounting is enabled. • **NM**—Non-configured MAC. • **R**—Remote PE MAC address is configured.
Logical interface	Name of the logical interface.
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 bridge domain in which the MAC address was learned.
VXLAN ID/VXLAN	VXLAN Network Identifier (VNI)
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning Tree Protocol epoch 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.

Sample Output

show bridge mac-table

```
user@host> 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 : default-switch
 Bridging domain : test1, VLAN : 1
   MAC                MAC      Logical          NH     RTR
   address            flags    interface        Index  ID
   01:00:0c:cc:cc:cc  S,NM     NULL
   01:00:0c:cc:cc:cd  S,NM     NULL
   01:00:0c:cd:cd:d0  S,NM     NULL
   64:87:88:6a:17:d0  D        ae0.1
   64:87:88:6a:17:f0  D        ae0.1
```

show bridge mac-table (with VXLAN enabled)

```
user@host> show bridge mac-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
 Bridging domain : vlan-1, VLAN : 1
  VXLAN: Id : 100, Multicast group: 226.1.1.1
   MAC                MAC      Logical
   address            flags    interface
   00:01:01:00:01:f7  D,SE     vtep.1052010
   00:03:00:32:01:f7  D,SE     vtep.1052011
   00:00:21:11:11:10  DL       ge-1/0/0.0
   00:00:21:11:11:11  DL       ge-1/1/0.0


Routing instance : default-switch
 Bridging domain : vlan-2, VLAN : 2, VXLAN : 200
  VXLAN: Id : 200, Multicast group: 226.1.1.2
   MAC                MAC      Logical
   address            flags    interface
   00:02:01:33:01:f7  D,SE     vtep.1052010
   00:04:00:14:01:f7  D,SE     vtep.1052011
   00:00:21:11:21:10  DL       ge-1/0/0.1
   00:00:21:11:21:11  DL       ge-1/1/0.1
```

show bridge mac-table count

```
user@host> show bridge mac-table count
2 MAC address learned in routing instance vs1 bridge domain vlan100

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ge-11/0/3.0                      1
    ge-11/1/4.100                    0
    ge-11/1/1.100                    0
    ge-11/1/0.100                    0
    xe-10/2/0.100                    1
    xe-10/0/0.100                    0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    2

0 MAC address learned in routing instance vs1 bridge domain vlan200

  MAC address count per interface within routing instance:
    Logical interface        MAC count
```

```
    ge-11/1/0.200                   0
    ge-11/1/1.200                   0
    ge-11/1/4.200                   0
    xe-10/0/0.200                   0
    xe-10/2/0.200                   0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                   0
```

show bridge mac-table detail

```
user@host> show bridge mac-table detail
MAC address: 00:00:00:19:1c:db
  Routing instance: vs1
   Bridging domain: vlan100
   Learning interface: ge-11/0/3.0     Learning VLAN: 0
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 4                            Sequence number: 0
   Learning mask: 0x800                IPC generation: 0

MAC address: 00:00:00:59:3a:2f
  Routing instance: vs1
   Bridging domain: vlan100
   Learning interface: xe-10/2/0.100   Learning VLAN: 0
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 7                            Sequence number: 0
   Learning mask: 0x400                IPC generation: 0
```

show ovsdb controller

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax
show ovsdb controller
<address *ip-address*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Display information and connection status for VMware NSX controllers to which the Juniper Networks device is connected. This command displays information for NSX controllers with connections to a Juniper Networks device that are made in the following ways:

- With explicit configuration—The connection is explicitly configured using the Junos OS CLI.
- Without explicit configuration—An NSX controller to which the Juniper Networks device is connected pushes information about other controllers in the same cluster to the device. With this method, the Juniper Networks device learns about the other controllers in the same cluster and connections to these controllers are established without explicit configuration.

Options
none—Display information about all NSX controllers to which the Juniper Networks device is connected.

address *ip-address*—(Optional) Display information about the NSX controller at the specified IP address.

Required Privilege Level admin

Related Documentation
- Setting Up the Open vSwitch Database Management Protocol on Juniper Networks Devices on page 81
- Understanding How to Set Up Open vSwitch Database Connections Between Juniper Networks Devices and Controllers on page 7

List of Sample Output
show ovsdb controller on page 113
show ovsdb controller address on page 113

Output Fields Table 13 on page 112 lists the output fields for the **show ovsdb controller** command. Output fields are listed in the approximate order in which they appear.

Table 13: show ovsdb controller Output Fields

Field Name	Field Descriptions
Controller IP address	IP address of an NSX controller to which the Juniper Networks device is connected.

Table 13: show ovsdb controller Output Fields *(continued)*

Field Name	Field Descriptions
Controller protocol	Protocol used by the Juniper Networks device to initiate a connection with an NSX controller.
Controller port	NSX controller port to which the Juniper Networks device is connected.
Controller connection	State of the connection between the Juniper Networks device and an NSX controller.
Controller seconds-since-connect	Number of seconds since the connection between the Juniper Networks device and NSX controller was established.
Controller seconds-since-disconnect	Number of seconds since the connection between the Juniper Networks device and NSX controller was dropped.
Controller connection status	Status of the connection between the Juniper Networks device and an NSX controller.

Sample Output

show ovsdb controller

```
user@host> show ovsdb controller
VTEP controller information:
Controller IP address: 10.168.66.189
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 56290
Controller seconds-since-disconnect: 0
Controller connection status: active

Controller IP address: 10.168.181.54
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 56292
Controller seconds-since-disconnect: 0
Controller connection status: active

Controller IP address: 10.168.182.45
Controller protocol: ssl
Controller port: 6632
Controller connection: up
Controller seconds-since-connect: 56292
Controller seconds-since-disconnect: 0
Controller connection status: active
```

show ovsdb controller address

```
user@host> show ovsdb controller address 10.168.182.45
VTEP controller information:
Controller IP address: 192.168.182.45
Controller protocol: ssl
Controller port: 6632
Controller connection: up
```

```
Controller seconds-since-connect: 56347
Controller seconds-since-disconnect: 0
Controller connection status: active
```

show ovsdb interface

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax
show ovsdb interface
<*interface-name*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Display information about Open vSwitch Database (OVSDB)-managed interfaces configured by using the **interfaces** ***interface-name*** statement in the [**edit protocols ovsdb**] hierarchy.

Options
none—Display information about all OVSDB-managed interfaces.

interface-name—(Optional) Display information about the specified OVSDB-managed interface.

Required Privilege Level admin

Related Documentation
- Configuring OVSDB-Managed VXLANs on page 84
- show ovsdb statistics interface on page 123

List of Sample Output
show ovsdb interface on page 115
show ovsdb (Specific Interface) on page 116

Output Fields Table 14 on page 115 lists the output fields for the **show ovsdb interface** command. Output fields are listed in the approximate order in which they appear.

Table 14: show ovsdb interface Output Fields

Field Name	Field Description
Interface	Name of interface.
VLAN ID	ID of Virtual Extensible LAN (VXLAN) with which the interface is associated. NOTE: MX Series routers and EX9200 switches do not support this field.
Bridge domain or VLAN	Bridge domain or VLAN under which the VXLAN is created. NOTE: MX Series routers and EX9200 switches do not support this field.

Sample Output

show ovsdb interface

```
user@host> show ovsdb interface
```

```
Interface               VLAN ID         Bridge-domain
ge-7/0/9.0
ge-7/0/9.1
irb.11
irb.12
irb.2
irb.3
xe-10/3/0.0
xe-10/3/0.1
```

show ovsdb (Specific Interface)

```
user@host> show ovsdb interface ge-7/0/9.0
Interface               VLAN ID         Bridge-domain
ge-7/0/9.0
```

show ovsdb logical-switch

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax
show ovsdb logical-switch
<*logical-switch-name*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description
Display information about logical switches, which you configured in NSX Manager or in the NSX API, and the corresponding Virtual Extensible LANs (VXLANs), which were configured on the Juniper Networks device.

In the command output, each logical switch is identified by a universally unique identifier (UUID), which in the context of this command, is also known as a logical switch name.

The **show ovsdb logical-switch** command displays the state of the logical switch (**Flags**), which can be one of the following:

Created by Controller—A logical switch was configured in NSX Manager. In this state, the logical switch and corresponding VXLAN are not yet operational.

Created by L2ALD—A VXLAN was configured on a Juniper Networks device. In this state, the logical switch-VXLAN are not yet operational.

Created by both—A logical switch was configured in NSX Manager, and a corresponding VXLAN was configured on a Juniper Networks device. In this state, the logical switch-VXLAN are operational.

Tunnel key mismatch—The VNIs specified in the logical switch and corresponding VXLAN configurations do not match. In this state, the logical switch-VXLAN are not yet operational.

For more information about configuring the logical switch and corresponding VXLAN, see "Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment" on page 10.

Options
none—Display information about all logical switches that are present in the OVSDB schema for physical devices.

logical-switch-name—(Optional) Display information about the specified logical switch.

Required Privilege Level
admin

Related Documentation
- Open vSwitch Database Schema For Physical Devices on page 12
- Troubleshooting a Nonoperational VMware NSX Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN on page 147

List of Sample Output show ovsdb logical-switch on page 118
show ovsdb logical-switch (Specific Logical Switch) on page 118

Output Fields Table 15 on page 118 lists the output fields for the **show ovsdb logical-switch** command. Output fields are listed in the approximate order in which they appear.

Table 15: show ovsdb logical-switch Output Fields

Field Name	Field Description
Logical Switch Name	UUID that is automatically generated by NSX and assigned to the logical switch after you configure it in NSX Manager or the NSX API. When configuring the corresponding VXLAN in the Junos OS CLI, the same UUID must be specified as the VXLAN name.
Flags	State of the logical switch. For possible states, see the Description section of this topic.
VNI	VNI that is configured for the logical switch and corresponding VXLAN.
Num of Remote MAC	The total number of remote MAC addresses associated with the logical switch. These addresses are learned by software and hardware virtual tunnel endpoints (VTEPs) in the NSX environment.
Num of Local MAC	The total number of local MAC addresses associated with the logical switch. *Local MAC addresses* are addresses learned on the local physical ports.

Sample Output

show ovsdb logical-switch

```
user@host> show ovsdb logical-switch
Logical switch information:
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
Flags: Created by both
VNI: 3
Num of Remote MAC: 13
Num of Local MAC: 12
Logical Switch Name: 9b4f880e-dac8-4612-a832-97ad9dec270f
Flags: Created by Controller
VNI: 50
Num of Remote MAC: 0
Num of Local MAC: 0
Logical Switch Name: bc0da2da-6c16-44bf-b655-442484294ded
Flags: Created by Controller
VNI: 51
Num of Remote MAC: 0
Num of Local MAC: 0
```

show ovsdb logical-switch (Specific Logical Switch)

```
user@host> show ovsdb logical-switch 24a76aff-7e61-4520-a78d-3eca26ad7510
Logical switch information:
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
Flags: Created by both
VNI: 3
Num of Remote MAC: 13
Num of Local MAC: 12
```


show ovsdb mac

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax
show ovsdb mac
<address *mac-address*>
<local>
<logical-switch *logical-switch-uuid*>
<multicast>
<remote>
<unicast>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Display MAC addresses, as well as information about the MAC addresses, learned by a Juniper Networks device that functions as a hardware virtual tunnel endpoint (VTEP). Using the Open vSwitch Database (OVSDB) management protocol, this hardware VTEP can learn about MAC addresses directly or from other software or hardware VTEPs. The MAC addresses learned directly by the hardware VTEP are known as local addresses, while the addresses learned from other software or hardware VTEPs are known as remote addresses.

Options Use one or more of the following options to display a more specific list of MAC addresses and information about the MAC addresses. For example, to display a list of local unicast MAC addresses, you can issue the **show ovsdb mac local unicast** command.

none—Display all MAC addresses, which includes all local, remote, unicast, and multicast addresses associated with all logical switches.

address *mac-address*—(Optional) Display the specified MAC address.

local—(Optional) Display all local MAC addresses.

logical-switch *logical-switch-uuid*—(Optional) Display all MAC addresses associated with the specified logical switch.

multicast—(Optional) Display all multicast MAC addresses.

remote—(Optional) Display all remote MAC addresses.

unicast—(Optional) Display all unicast MAC addresses.

Required Privilege Level admin

Related Documentation
- Open vSwitch Database Schema For Physical Devices on page 12

List of Sample Output
show ovsdb mac on page 120
show ovsdb mac address on page 121

show ovsdb mac logical-switch on page 121
show ovsdb mac local unicast on page 121

Output Fields Table 16 on page 120 lists the output fields for the **show ovsdb mac** command. Output fields are listed in the approximate order in which they appear.

Table 16: show ovsdb mac Output Fields

Field Name	Field Description
Logical Switch Name	Universally unique identifier (UUID) of the logical switch. For more information about logical switches and UUIDs, see "Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment" on page 10.
MAC Address	MAC addresses of virtual machines (VMs).
IP Address	IP address of VMs. NOTE: If the IP addresses of VMs are not published by the NSX controller, this field displays **0.0.0.0**.
Encapsulation	Encapsulation type.
VTEP Address	IP address of the hardware or software VTEP from which the MAC address was learned. Further, this VTEP can forward VM traffic to the associated host.

Sample Output

show ovsdb mac

```
user@host> show ovsdb mac
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
  Mac                 IP                Encapsulation      Vtep
  Address             Address                              Address
  02:00:00:00:03:01   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:02   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:03   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:04   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  02:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  04:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:01   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:02   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:03   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:04   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  06:00:00:00:03:05   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  40:b4:f0:06:6f:f0   0.0.0.0           Vxlan over Ipv4    10.255.18.22
  ff:ff:ff:ff:ff:ff   0.0.0.0           Vxlan over Ipv4    10.100.100.1

Logical Switch Name: bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab
  Mac                 IP                Encapsulation      Vtep
  Address             Address                              Address
  02:00:00:00:11:01   0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:02   0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:03   0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:04   0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:05   0.0.0.0           Vxlan over Ipv4    10.1.1.29
  04:00:00:00:11:05   0.0.0.0           Vxlan over Ipv4    10.1.1.29
```

```
  06:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  40:b4:f0:06:6f:f0      0.0.0.0           Vxlan over Ipv4    10.1.1.29
  00:23:9c:5e:a7:f0      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:01      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:02      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:03      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:04      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:05      0.0.0.0           Vxlan over Ipv4    10.255.18.22
  ff:ff:ff:ff:ff:ff      0.0.0.0           Vxlan over Ipv4    10.110.110.1
...
```

show ovsdb mac address

```
user@host> show ovsdb mac address 02:00:00:00:03:01

  Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  02:00:00:00:03:01     0.0.0.0           Vxlan over Ipv4    10.255.18.22
```

show ovsdb mac logical-switch

```
user@host> show ovsdb mac logical-switch bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab
Logical Switch Name: bf6d4fd4-f5f6-430c-8c37-4033ef1c55ab
  Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  02:00:00:00:11:01     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:02     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:03     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:04     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  02:00:00:00:11:05     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  04:00:00:00:11:05     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:01     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:02     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:03     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:04     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  06:00:00:00:11:05     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  40:b4:f0:06:6f:f0     0.0.0.0           Vxlan over Ipv4    10.1.1.29
  00:23:9c:5e:a7:f0     0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:01     0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:02     0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:03     0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:04     0.0.0.0           Vxlan over Ipv4    10.255.18.22
  08:00:00:00:11:05     0.0.0.0           Vxlan over Ipv4    10.255.18.22
  ff:ff:ff:ff:ff:ff     0.0.0.0           Vxlan over Ipv4    10.110.110.1
```

show ovsdb mac local unicast

```
user@host> show ovsdb mac local unicast
Logical Switch Name: 24a76aff-7e61-4520-a78d-3eca26ad7510
  Mac                   IP                Encapsulation      Vtep
  Address               Address                              Address
  02:00:00:00:03:01     0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:02     0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:03     0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:04     0.0.0.0           Vxlan over Ipv4    10.255.181.72
  02:00:00:00:03:05     0.0.0.0           Vxlan over Ipv4    10.255.181.72
  04:00:00:00:03:05     0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:01     0.0.0.0           Vxlan over Ipv4    10.255.181.72
```

```
  06:00:00:00:03:02      0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:03      0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:04      0.0.0.0           Vxlan over Ipv4    10.255.181.72
  06:00:00:00:03:05      0.0.0.0           Vxlan over Ipv4    10.255.181.72
  40:b4:f0:06:6f:f0      0.0.0.0           Vxlan over Ipv4    10.255.181.72
...
```

show ovsdb statistics interface

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax
show ovsdb statistics interface
<*interface-name*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Display statistics for Open vSwitch Database (OVSDB)-managed interfaces configured by using the **interfaces** ***interface-name*** statement in the [**edit protocols ovsdb**] hierarchy.

When an interface is configured as OVSDB-managed, the collection of statistics for that interface begins, and the statistics displayed at any given time reflects the data collected up to that point.

Options **none**—Display statistics for all configured OVSDB-managed interfaces.

interface-name—(Optional) Display statistics for the specified interface.

Required Privilege Level admin

Related Documentation
- interfaces on page 91

List of Sample Output
show ovsdb statistics interface on page 123
show ovsdb statistics interface (Specific Interface) on page 124

Output Fields Table 17 on page 123 lists the output fields for the **show ovsdb statistics interface** command. Output fields are listed in the approximate order in which they appear.

Table 17: show ovsdb statistics interface Output Fields

Field Name	Field Descriptions
Num of rx pkts	Number of packets received by the interface.
Num of tx pkts	Number of packets sent by the interface.
Num of rx bytes	Number of bytes received by the interface.
Num of tx bytes	Number of bytes sent by the interface.

Sample Output

show ovsdb statistics interface

```
user@host> show ovsdb statistics interface
```

```
Interface Name: ge-7/0/9.0
Num of rx pkts: 945                    Num of tx pkts: 113280890
Num of rx bytes: 56700                  Num of tx bytes: 57531319540
Interface Name: ge-7/0/10.0
Num of rx pkts: 459                    Num of tx pkts: 473840856
Num of rx bytes: 84747                  Num of tx bytes: 45830738532
Interface Name: ge-7/0/11.0
Num of rx pkts: 305                    Num of tx pkts: 367483456
Num of rx bytes: 98974                  Num of tx bytes: 33495468092
```

show ovsdb statistics interface (Specific Interface)

```
user@host> show ovsdb statistics interface ge-7/0/9.0
Interface Name: ge-7/0/9.0
Num of rx pkts: 945                    Num of tx pkts: 113280890
Num of rx bytes: 56700                  Num of tx bytes: 57531319540
```

show ovsdb virtual-tunnel-end-point

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Syntax
show ovsdb virtual-tunnel-end-point
<address *ip-address*>
<encapsulation <*encapsulation-type*>

Release Information
Command introduced in Junos OS Release 14.1R2.
Command introduced in Junos OS Release 14.1X53-D10 for QFX Series switches.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Display information about the following entities that the Juniper Networks device has learned:

- Other Juniper Networks devices that function as hardware virtual tunnel endpoints (VTEPs)
- Software VTEPs
- Service nodes

Options
none—Display information about all VTEPs and service nodes that the Juniper Networks device has learned.

address *ip-address*—(Optional) Display information about the entity with specified IP address.

encapsulation *encapsulation-type*—(Optional) Display information about all entities with the specified encapsulation type.

Required Privilege Level admin

List of Sample Output
show ovsdb virtual-tunnel-end-point on page 126
show ovsdb virtual-tunnel-end-point address (Specific Address) on page 126
show ovsdb virtual-tunnel-end-point encapsulation (Specific Encapsulation) on page 126
show ovsdb virtual-tunnel-end-point address (Specific Address) encapsulation (Specific Encapsulation) on page 126

Output Fields Table 18 on page 125 lists the output fields for the **show ovsdb virtual-tunnel-end-point** command. Output fields are listed in the approximate order in which they appear.

Table 18: show ovsdb virtual-tunnel-end-point Output Fields

Field Name	Field Description
Encapsulation	Encapsulation type of entity.
IP Address	IP address of entity.
Num of MACs	Number of MAC addresses learned by the entity.

Sample Output

show ovsdb virtual-tunnel-end-point

```
user@host> show ovsdb virtual-tunnel-end-point
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
VXLAN over IPv4            10.255.181.50                   12
VXLAN over IPv4            10.255.181.72                   24
```

show ovsdb virtual-tunnel-end-point address (Specific Address)

```
user@host> show ovsdb virtual-tunnel-end-point address 10.255.181.43
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
```

show ovsdb virtual-tunnel-end-point encapsulation (Specific Encapsulation)

```
user@host> show ovsdb virtual-tunnel-end-point encapsulation vxlan-over-ipv4
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
VXLAN over IPv4            10.255.181.50                   12
VXLAN over IPv4            10.255.181.72                   24
```

show ovsdb virtual-tunnel-end-point address (Specific Address) encapsulation (Specific Encapsulation)

```
user@host> show ovsdb virtual-tunnel-end-point address 10.255.181.43 encapsulation
vxlan-over-ipv4
Encapsulation              Ip Address                      Num of MAC's
VXLAN over IPv4            10.255.181.43                   24
```

show vpls mac-table

Supported Platforms MX Series

Syntax
show vpls mac-table
<**brief | detail | extensive | summary**>
<**bridge-domain** *bridge-domain-name*>
<**instance** *instance-name*>
<**interface** *interface-name*>
<**logical-system (all |** *logical-system-name*)>
<*mac-address*>
<**vlan-id** *vlan-id-number*>

Release Information Command introduced in Junos OS Release 8.5.

Description Display learned VPLS MAC address information.

Options
none—Display all learned VPLS MAC address information.

brief | detail | extensive | summary—(Optional) Display the specified level of output.

bridge-domain ***bridge-domain-name***—(Optional) Display learned VPLS MAC addresses for the specified bridge domain.

instance ***instance-name***—(Optional) Display learned VPLS MAC addresses for the specified instance.

interface ***interface-name***—(Optional) Display learned VPLS MAC addresses for the specified instance.

logical-system (all | ***logical-system-name*****)**—(Optional) Display learned VPLS MAC addresses for all logical systems or for the specified logical system.

mac-address—(Optional) Display the specified learned VPLS MAC address information..

vlan-id ***vlan-id-number***—(Optional) Display learned VPLS MAC addresses for the specified VLAN.

Required Privilege Level view

List of Sample Output
show vpls mac-table on page 128
show vpls mac-table (with VXLAN enabled) on page 129
show vpls mac-table count on page 129
show vpls mac-table detail on page 130
show vpls mac-table extensive on page 130

Output Fields Table 19 on page 128 describes the output fields for the **show bridge mac-table** command. Output fields are listed in the approximate order in which they appear.

Table 19: show vpls mac-table Output fields

Field Name	Field Description
Routing instance	Name of the routing instance.
Bridging domain	Name of the bridging domain.
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 configured. • **D**—Dynamic MAC address learned. • **SE**—MAC accounting is enabled. • **NM**—Nonconfigured MAC.
Logical interface	Name of the logical interface.
MAC count	Number of MAC addresses learned on a specific routing instance or interface.
Learning interface	Logical interface or logical Label Switched Interface (LSI) the address is learned on.
Base learning interface	Base learning interface of the MAC address. This field is introduced in Junos OS Release 14.2.
Learn VLAN ID/VLAN	VLAN ID of the routing instance or bridge domain in which the MAC address was learned.
VXLAN ID/VXLAN	VXLAN Network Identifier (VNI)
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning Tree Protocol epoch 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 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.

Sample Output

show vpls mac-table

```
user@host> show vpls mac-table
MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_ldp1
VLAN : 223
   MAC                 MAC      Logical
   address             flags    interface
   00:90:69:9c:1c:5d   D        ge-0/2/5.400
```

```
MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_red
VLAN : 401
   MAC                 MAC      Logical
   address             flags    interface
   00:00:aa:12:12:12   D        lsi.1051138
   00:05:85:74:9f:f0   D        lsi.1051138
```

show vpls mac-table (with VXLAN enabled)

```
user@host> show vpls mac-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 : vpls_4site:1000
 Bridging domain : __vpls_4site:1000__, VLAN : 4094,4093
  VXLAN: Id : 300, Multicast group: 226.1.1.3
   MAC                 MAC      Logical
   address             flags    interface
   00:01:01:00:01:f4   D,SE     ge-4/2/0.1000
   00:02:01:33:01:f4   D,SE     lsi.1052004
   00:03:00:32:01:f4   D,SE     lsi.1048840
   00:04:00:14:01:f4   D,SE     lsi.1052005
   00:02:01:33:02:f7   D,SE     vtep.1052010
   00:04:00:14:02:f7   D,SE     vtep.1052011
```

show vpls mac-table count

```
user@host> show vpls mac-table count
0 MAC address learned in routing instance __juniper_private1__

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    lc-0/0/0.32769                   0
    lc-0/1/0.32769                   0
    lc-0/2/0.32769                   0
    lc-2/0/0.32769                   0
    lc-0/3/0.32769                   0
    lc-2/1/0.32769                   0
    lc-9/0/0.32769                   0
    lc-11/0/0.32769                  0
    lc-2/2/0.32769                   0
    lc-9/1/0.32769                   0
    lc-11/1/0.32769                  0
    lc-2/3/0.32769                   0
    lc-9/2/0.32769                   0
    lc-11/2/0.32769                  0
    lc-11/3/0.32769                  0
    lc-9/3/0.32769                   0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    0

1 MAC address learned in routing instance vpls_ldp1

  MAC address count per interface within routing instance:
    Logical interface        MAC count
```

```
    lsi.1051137                    0
    ge-0/2/5.400                   1

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                 0                   1

1 MAC address learned in routing instance vpls_red

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ge-0/2/5.300                   1

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                 0                   1
```

show vpls mac-table detail

```
user@host> show vpls mac-table detail
MAC address: 00:90:69:9c:1c:5d
  Routing instance: vpls_ldp1
   Learning interface: ge-0/2/5.400
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0

MAC address: 00:90:69:9c:1c:5d
  Routing instance: vpls_red
   Learning interface: ge-0/2/5.300
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0
```

show vpls mac-table extensive

```
user@host> show vpls mac-table extensive

MAC address: 00:10:00:01:00:00
  Routing instance: vpls_1
   Bridging domain: __vpls_1__, VLAN : NA
   Learning interface: lsi.1049165
   Base learning interface: lsi.1049165
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x00000001

MAC address: 00:10:00:01:00:01
  Routing instance: vpls_1
   Bridging domain: __vpls_1__, VLAN : NA
   Learning interface: lsi.1049165
   Base learning interface: lsi.1049165
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x00000001

MAC address: 00:10:00:01:00:02
  Routing instance: vpls_1
   Bridging domain: __vpls_1__, VLAN : NA
   Learning interface: lsi.1049165
   Base learning interface: lsi.1049165
```



```
    Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
    Epoch: 0                            Sequence number: 1
    Learning mask: 0x00000001

MAC address: 00:10:00:01:00:03
  Routing instance: vpls_1
    Bridging domain: __vpls_1__, VLAN : NA
    Learning interface: lsi.1049165
    Base learning interface: lsi.1049165
    Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
    Epoch: 0                            Sequence number: 1
    Learning mask: 0x00000001
```

CHAPTER 8

VXLAN Monitoring Commands

show bridge mac-table

Supported Platforms MX Series

Syntax
show bridge mac-table
<brief | count | detail | extensive>
<bridge-domain (all | *bridge-domain-name*)>
<global-count>
<interface *interface-name*>
<*mac-address*>
<vlan-id (all-vlan | *vlan-id*)>

Release Information Command introduced in Junos OS Release 8.4.

Description (MX Series routers only) Display Layer 2 MAC address information.

Options
none—Display all learned Layer 2 MAC address information.

brief | count | detail | extensive—(Optional) Display the specified level of output.

bridge-domain (all | *bridge-domain-name*)—(Optional) Display learned Layer 2 MAC addresses for all bridging domains or for the specified bridging domain.

global-count—(Optional) Display the total number of learned Layer 2 MAC addresses on the system.

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.

mac-address—(Optional) Display the specified learned Layer 2 MAC address information.

vlan-id (all-vlan | *vlan-id*)—(Optional) Display learned Layer 2 MAC addresses for all VLANs or for the specified VLAN.

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

List of Sample Output
show bridge mac-table on page 135
show bridge mac-table (with VXLAN enabled) on page 136
show bridge mac-table count on page 136
show bridge mac-table detail on page 137

Output Fields Table 12 on page 109 describes the output fields for the **show bridge mac-table** command. Output fields are listed in the approximate order in which they appear.

Table 20: show bridge mac-table Output fields

Field Name	Field Description
Routing instance	Name of the routing instance.
Bridging domain	Name of the bridging domain.
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. • **C**—Control MAC address is configured. • **SE**—MAC accounting is enabled. • **NM**—Non-configured MAC. • **R**—Remote PE MAC address is configured.
Logical interface	Name of the logical interface.
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 bridge domain in which the MAC address was learned.
VXLAN ID/VXLAN	VXLAN Network Identifier (VNI)
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning Tree Protocol epoch 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.

Sample Output

show bridge mac-table

```
user@host> 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 : default-switch
 Bridging domain : test1, VLAN : 1
   MAC                MAC      Logical          NH     RTR
   address            flags    interface        Index  ID
   01:00:0c:cc:cc:cc  S,NM     NULL
   01:00:0c:cc:cc:cd  S,NM     NULL
   01:00:0c:cd:cd:d0  S,NM     NULL
   64:87:88:6a:17:d0  D        ae0.1
   64:87:88:6a:17:f0  D        ae0.1
```

show bridge mac-table (with VXLAN enabled)

```
user@host> show bridge mac-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
 Bridging domain : vlan-1, VLAN : 1
  VXLAN: Id : 100, Multicast group: 226.1.1.1
   MAC                MAC      Logical
   address            flags    interface
   00:01:01:00:01:f7  D,SE     vtep.1052010
   00:03:00:32:01:f7  D,SE     vtep.1052011
   00:00:21:11:11:10  DL       ge-1/0/0.0
   00:00:21:11:11:11  DL       ge-1/1/0.0


Routing instance : default-switch
 Bridging domain : vlan-2, VLAN : 2, VXLAN : 200
  VXLAN: Id : 200, Multicast group: 226.1.1.2
   MAC                MAC      Logical
   address            flags    interface
   00:02:01:33:01:f7  D,SE     vtep.1052010
   00:04:00:14:01:f7  D,SE     vtep.1052011
   00:00:21:11:21:10  DL       ge-1/0/0.1
   00:00:21:11:21:11  DL       ge-1/1/0.1
```

show bridge mac-table count

```
user@host> show bridge mac-table count
2 MAC address learned in routing instance vs1 bridge domain vlan100

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ge-11/0/3.0                      1
    ge-11/1/4.100                    0
    ge-11/1/1.100                    0
    ge-11/1/0.100                    0
    xe-10/2/0.100                    1
    xe-10/0/0.100                    0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    2

0 MAC address learned in routing instance vs1 bridge domain vlan200

  MAC address count per interface within routing instance:
    Logical interface        MAC count
```

```
    ge-11/1/0.200                   0
    ge-11/1/1.200                   0
    ge-11/1/4.200                   0
    xe-10/0/0.200                   0
    xe-10/2/0.200                   0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                   0
```

show bridge mac-table detail

```
user@host> show bridge mac-table detail
MAC address: 00:00:00:19:1c:db
  Routing instance: vs1
   Bridging domain: vlan100
   Learning interface: ge-11/0/3.0     Learning VLAN: 0
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 4                            Sequence number: 0
   Learning mask: 0x800                IPC generation: 0

MAC address: 00:00:00:59:3a:2f
  Routing instance: vs1
   Bridging domain: vlan100
   Learning interface: xe-10/2/0.100   Learning VLAN: 0
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 7                            Sequence number: 0
   Learning mask: 0x400                IPC generation: 0
```

show vpls mac-table

Supported Platforms

Syntax
show vpls mac-table
<**brief | detail | extensive | summary**>
<**bridge-domain** *bridge-domain-name*>
<**instance** *instance-name*>
<**interface** *interface-name*>
<**logical-system (all |** *logical-system-name*)>
<*mac-address*>
<**vlan-id** *vlan-id-number*>

Release Information Command introduced in Junos OS Release 8.5.

Description Display learned VPLS MAC address information.

Options **none**—Display all learned VPLS MAC address information.

brief | detail | extensive | summary—(Optional) Display the specified level of output.

bridge-domain ***bridge-domain-name***—(Optional) Display learned VPLS MAC addresses for the specified bridge domain.

instance ***instance-name***—(Optional) Display learned VPLS MAC addresses for the specified instance.

interface ***interface-name***—(Optional) Display learned VPLS MAC addresses for the specified instance.

logical-system (all | ***logical-system-name*****)**—(Optional) Display learned VPLS MAC addresses for all logical systems or for the specified logical system.

mac-address—(Optional) Display the specified learned VPLS MAC address information..

vlan-id ***vlan-id-number***—(Optional) Display learned VPLS MAC addresses for the specified VLAN.

Required Privilege Level view

List of Sample Output
show vpls mac-table on page 139
show vpls mac-table (with VXLAN enabled) on page 140
show vpls mac-table count on page 140
show vpls mac-table detail on page 141
show vpls mac-table extensive on page 141

Output Fields Table 19 on page 128 describes the output fields for the **show bridge mac-table** command. Output fields are listed in the approximate order in which they appear.

Table 21: show vpls mac-table Output fields

Field Name	Field Description
Routing instance	Name of the routing instance.
Bridging domain	Name of the bridging domain.
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 configured. • **D**—Dynamic MAC address learned. • **SE**—MAC accounting is enabled. • **NM**—Nonconfigured MAC.
Logical interface	Name of the logical interface.
MAC count	Number of MAC addresses learned on a specific routing instance or interface.
Learning interface	Logical interface or logical Label Switched Interface (LSI) the address is learned on.
Base learning interface	Base learning interface of the MAC address. This field is introduced in Junos OS Release 14.2.
Learn VLAN ID/VLAN	VLAN ID of the routing instance or bridge domain in which the MAC address was learned.
VXLAN ID/VXLAN	VXLAN Network Identifier (VNI)
Layer 2 flags	Debugging flags signifying that the MAC address is present in various lists.
Epoch	Spanning Tree Protocol epoch 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 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.

Sample Output

show vpls mac-table

```
user@host> show vpls mac-table
MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_ldp1
VLAN : 223
   MAC                 MAC      Logical
   address             flags    interface
   00:90:69:9c:1c:5d   D        ge-0/2/5.400
```

```
MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_red
VLAN : 401
   MAC                 MAC      Logical
   address             flags    interface
   00:00:aa:12:12:12   D        lsi.1051138
   00:05:85:74:9f:f0   D        lsi.1051138
```

show vpls mac-table (with VXLAN enabled)

```
user@host> show vpls mac-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 : vpls_4site:1000
 Bridging domain : __vpls_4site:1000__, VLAN : 4094,4093
  VXLAN: Id : 300, Multicast group: 226.1.1.3
   MAC                 MAC      Logical
   address             flags    interface
   00:01:01:00:01:f4   D,SE     ge-4/2/0.1000
   00:02:01:33:01:f4   D,SE     lsi.1052004
   00:03:00:32:01:f4   D,SE     lsi.1048840
   00:04:00:14:01:f4   D,SE     lsi.1052005
   00:02:01:33:02:f7   D,SE     vtep.1052010
   00:04:00:14:02:f7   D,SE     vtep.1052011
```

show vpls mac-table count

```
user@host> show vpls mac-table count
0 MAC address learned in routing instance __juniper_private1__

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    lc-0/0/0.32769                   0
    lc-0/1/0.32769                   0
    lc-0/2/0.32769                   0
    lc-2/0/0.32769                   0
    lc-0/3/0.32769                   0
    lc-2/1/0.32769                   0
    lc-9/0/0.32769                   0
    lc-11/0/0.32769                  0
    lc-2/2/0.32769                   0
    lc-9/1/0.32769                   0
    lc-11/1/0.32769                  0
    lc-2/3/0.32769                   0
    lc-9/2/0.32769                   0
    lc-11/2/0.32769                  0
    lc-11/3/0.32769                  0
    lc-9/3/0.32769                   0

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    0

1 MAC address learned in routing instance vpls_ldp1

  MAC address count per interface within routing instance:
    Logical interface        MAC count
```

```
    lsi.1051137                    0
    ge-0/2/5.400                   1

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    1

1 MAC address learned in routing instance vpls_red

  MAC address count per interface within routing instance:
    Logical interface        MAC count
    ge-0/2/5.300                   1

  MAC address count per learn VLAN within routing instance:
    Learn VLAN ID            MAC count
                0                    1
```

show vpls mac-table detail

```
user@host> show vpls mac-table detail
MAC address: 00:90:69:9c:1c:5d
  Routing instance: vpls_ldp1
   Learning interface: ge-0/2/5.400
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0

MAC address: 00:90:69:9c:1c:5d
  Routing instance: vpls_red
   Learning interface: ge-0/2/5.300
   Layer 2 flags: in_ifd, in_ifl, in_vlan, kernel
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x1                IPC generation: 0
```

show vpls mac-table extensive

```
user@host> show vpls mac-table extensive

MAC address: 00:10:00:01:00:00
  Routing instance: vpls_1
   Bridging domain: __vpls_1__, VLAN : NA
   Learning interface: lsi.1049165
   Base learning interface: lsi.1049165
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x00000001

MAC address: 00:10:00:01:00:01
  Routing instance: vpls_1
   Bridging domain: __vpls_1__, VLAN : NA
   Learning interface: lsi.1049165
   Base learning interface: lsi.1049165
   Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
   Epoch: 0                          Sequence number: 1
   Learning mask: 0x00000001

MAC address: 00:10:00:01:00:02
  Routing instance: vpls_1
   Bridging domain: __vpls_1__, VLAN : NA
   Learning interface: lsi.1049165
   Base learning interface: lsi.1049165
```

```
    Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
    Epoch: 0                          Sequence number: 1
    Learning mask: 0x00000001

MAC address: 00:10:00:01:00:03
  Routing instance: vpls_1
    Bridging domain: __vpls_1__, VLAN : NA
    Learning interface: lsi.1049165
    Base learning interface: lsi.1049165
    Layer 2 flags: in_hash,in_ifd,in_ifl,in_vlan,in_rtt,kernel,in_ifbd
    Epoch: 0                          Sequence number: 1
    Learning mask: 0x00000001
```

Verifying VXLAN Reachability

Supported Platforms QFX Series standalone switches

Purpose On the local VTEP, verify that there is connectivity with the remote VTEP.

Action

```
user@switch> show ethernet-switching vxlan-tunnel-end-point remote
Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
<default>                 0   10.1.1.2     lo0.0    0
RVTEP-IP         IFL-Idx   NH-Id
10.1.1.2         559       1728
VNID             MC-Group-IP
100              232.1.1.1
```

Meaning The remote VTEP is reachable because its IP address appears in the output. The output also shows that the VXLAN (VNI 100) and corresponding multicast group are configured correctly on the remote VTEP.

Related Documentation

- Understanding VXLANs on page 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

Verifying That a Local VXLAN VTEP is Configured Correctly

Supported Platforms QFX Series standalone switches

Purpose Verify that a local VTEP is correct..

Action

```
user@switch> show ethernet-switching vxlan-tunnel-end-point source

Logical System Name       Id  SVTEP-IP         IFL   L3-Idx
<default>                 0   10.1.1.1     lo0.0    0
L2-RTT                  Bridge Domain              VNID     MC-Group-IP
default-switch          VLAN1+100                  100      232.1.1.1
```

Meaning The output should show the correct tunnel source IP address (loopback address), VLAN, and multicast group for the VXLAN.

Related Documentation

- Understanding VXLANs on page 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

Verifying MAC Learning from a Remote VTEP

Supported Platforms QFX Series standalone switches

Purpose Verify that a local VTEP is learning MAC addresses from a remote VTEP.

Action **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)
Ethernet switching table : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
    name                address             flags              interface
    VLAN1                00:00:00:ff:ff:ff   D             -    vtep.12345
    VLAN1                00:10:94:00:00:02   D         -        xe-0/0/0.0
```

Meaning This shows the MAC addresses learned from the remote VTEP (in addition to those learned on the normal Layer 2 interfaces). It also shows the logical name of the remote VTEP interface (**vtep.12345** in the above output).

Related Documentation

- Understanding VXLANs on page 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

Monitor a Remote VTEP Interface

Supported Platforms QFX Series standalone switches

Purpose Monitor traffic details for a remote VTEP interface.

Action user@switch> show interface *logical-name* detail

```
M    Flags: Up SNMP-Traps Encapsulation: ENET2
    VXLAN Endpoint Type: Remote, VXLAN Endpoint Address: 10.1.1.2, L2 Routing
Instance: default-switch, L3 Routing Instance: default
    Traffic statistics:
     Input  bytes  :         228851738624
     Output bytes  :                    0
     Input  packets:            714162415
     Output packets:                    0
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Transit statistics:
     Input  bytes  :         228851738624                  0 bps
     Output bytes  :                    0                  0 bps
     Input  packets:            714162415                  0 pps
     Output packets:                    0                  0 pps
    Protocol eth-switch, MTU: 1600, Generation: 277, Route table: 5
```

Meaning This shows traffic details for the remote VTEP interface. To get this information, you must supply the logical name of the remote VTEP interface (vtep.12345 in the above output), which you can learn by using the **show ethernet-switching table** command.

Related Documentation

- Understanding VXLANs on page 15
- *Configuring VXLANs on a QFX5100 Switch*
- *Examples: Configuring VXLANs on QFX Series Switches*

PART 4

Troubleshooting

CHAPTER 9

Troubleshooting Procedures

Troubleshooting a Nonoperational VMware NSX Logical Switch and Corresponding Junos OS OVSDB-Managed VXLAN

Supported Platforms EX Series, MX Series, QFX Series standalone switches

Problem **Description:** A logical switch configured by using VMware NSX Manager or the NSX API, and the corresponding Open vSwitch Database (OVSDB)-managed Virtual Extensible LAN (VXLAN), which is configured on a Juniper Networks device, are not exchanging MAC addresses learned in the NSX and Junos OS environments, respectively. Also, the **Flags** field in the **show ovsdb logical-switch** operational mode command output is one of the following:

- **Created by Controller**
- **Created by L2ALD**
- **Tunnel key mismatch**

Cause

- If the **Flags** field displays **Created by Controller**, a logical switch is configured in NSX Manager or in the NSX API, but a corresponding VXLAN is not configured or is improperly configured on the Juniper Networks device.
- If the **Flags** field displays **Created by L2ALD**, a VXLAN is configured on the Juniper Networks device, but a corresponding logical switch is not configured in NSX Manager or in the NSX API.
- If the **Flags** field displays **Tunnel key mismatch**, the VXLAN network identifiers (VNIs) in the logical switch and corresponding VXLAN configurations do not match.

Solution If the **Flags** field displays **Created by Controller**, determine whether a corresponding VXLAN is configured on the device. If the VXLAN is not configured, configure it using the procedure in "Configuring OVSDB-Managed VXLANs" on page 84. If a VXLAN is configured, check the VXLAN name to make sure that it is the same as the universally unique identifier (UUID) of the logical switch. Also, check the VNI to make sure that the value is the same as the value in the logical switch configuration.

If the **Flags** field displays **Created by L2ALD**, determine whether a corresponding logical switch is configured in NSX Manager or in the NSX API. If a logical switch is not configured, configure one, keeping in mind that NSX automatically generates a UUID for the logical switch and that this UUID must be used as the name of the VXLAN. That is, the VXLAN name must be reconfigured with the logical switch UUID. Another possibility is that the logical switch might exist, but the logical switch UUID might not be the VXLAN name. In NSX Manager or in the NSX API, check for a logical switch that has the same configuration as the VXLAN but has a different UUID.

If the **Flags** field displays **Tunnel key mismatch**, check the values of the VNI in NSX Manager or in the NSX API and the Junos OS CLI, and change the incorrect value.

Related Documentation

- Understanding How to Set Up Virtual Extensible LANs in an Open vSwitch Database Environment on page 10
- **show ovsdb logical-switch on page 117**