CONFIGURATION GUIDE

Extreme SLX-OS Layer 2 Switching Configuration Guide, 17r.2.01

Supporting the ExtremeRouting SLX 9850 and ExtremeSwitching SLX 9540 Devices

9035311-03 Rev AA
October 2019

Copyright © 2019 Extreme Networks, Inc. All Rights Reserved.

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, please see: www.extremenetworks.com/company/legal/trademarks

Open Source Declarations

Some software files have been licensed under certain open source or third-party licenses. End-user license agreements and open source declarations can be found at: www.extremenetworks.com/support/policies/software-licensing

Contents

Preface

This section discusses the conventions used in this guide, ways to provide feedback, additional help, and other Extreme Networks® publications.

Conventions

This section discusses the conventions used in this guide.

Notes, cautions, and warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A Note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

CAUTION
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used to highlight specific words or phrases.

Format	Description
bold text	Identifies command names.
	Identifies keywords and operands.
	Identifies the names of GUI elements.
	Identifies text to enter in the GUI.
italic text	Identifies emphasis.
	Identifies variables.
	Identifies document titles.

Format	Description
`Courier font`	Identifies CLI output. Identifies command syntax examples.

Command syntax conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention	Description
bold text	Identifies command names, keywords, and command options.
italic text	Identifies a variable.
[ ]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{ **x** | **y** | **z** }	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
x | **y**	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, for example, passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*[*member*...].
\	Indicates a "soft" line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Documentation and Training

To find Extreme Networks product guides, visit our documentation pages at:

Current Product Documentation	www.extremenetworks.com/documentation/
Archived Documentation (for earlier versions and legacy products)	www.extremenetworks.com/support/documentation-archives/
Release Notes	www.extremenetworks.com/support/release-notes
Hardware/Software Compatibility Matrices	https://www.extremenetworks.com/support/compatibility-matrices/
White papers, data sheets, case studies, and other product resources	https://www.extremenetworks.com/resources/

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For more information, visit www.extremenetworks.com/education/.

Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal	Search the GTAC (Global Technical Assistance Center) knowledge base, manage support cases and service contracts, download software, and obtain product licensing, training, and certifications.
The Hub	A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
Call GTAC	For immediate support: 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribing to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

1. Go to www.extremenetworks.com/support/service-notification-form.
2. Complete the form with your information (all fields are required).
3. Select the products for which you would like to receive notifications.

   NOTE
   You can modify your product selections or unsubscribe at any time.

4. Click **Submit**.

Providing Feedback to Us

Quality is our first concern at Extreme Networks, and we have made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you! We welcome all feedback but especially want to know about:

- Content errors or confusing or conflicting information.
- Ideas for improvements to our documentation so you can find the information you need faster.
- Broken links or usability issues.

If you would like to provide feedback to the Extreme Networks Information Development team, you can do so in two ways:

- Use our short online feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@extremenetworks.com.

Please provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

About This Document

Supported hardware and software

In those instances in which procedures or parts of procedures documented here apply to some devices but not to others, this guide identifies exactly which devices are supported and which are not.

Although many different software and hardware configurations are tested and supported by Extreme Networks for the current release, documenting all possible configurations and scenarios is beyond the scope of this document.

The following hardware platforms are supported by this release:

- ExtremeRouting SLX 9850-4
- ExtremeRouting SLX 9850-8
- ExtremeRouting SLX 9640

To obtain information about other releases, refer to the documentation specific to that release.

Interface module capabilities

The following table lists the supported capabilities for the following SLX 9850 interface modules:

- BR-SLX9850-10Gx72S-M
- BR-SLX9850-100Gx36CQ-M
- BR-SLX9850-10Gx72S-D
- BR-SLX9850-100Gx36CQ-D
- BR-SLX9850-100Gx12CQ-M

TABLE 1 SLX 9850 interface modules capabilities

Capability	Modular interface module
MPLS	Yes
Packet Buffer memory per interface module	12GB (BR-SLX9850-10Gx72S-M) 36GB (BR-SLX9850-100Gx36CQ-M) 8GB (BR-SLX9850-10Gx72S-D) 24GB (BR-SLX9850-100Gx36CQ-D) 8GB (BR-SLX9850-100Gx12CQ-M)

What's new in this document

On October 30, 2017, Extreme Networks, Inc. acquired the SLX-OS product line from Brocade Communications Systems, Inc. This document has been updated to remove or replace all references to Brocade Communications, Inc. with Extreme Networks., Inc., as appropriate

The following table describes new information added to this guide for the SLX OS 17r.2.01 software release.

TABLE 2 Summary of enhancements in this SLX OS release

Feature	Description	Described in
Logical VTEP	A logical VXLAN tunnel end point (LVTEP) is supported at Layer 2.	VXLAN Layer 2 Gateway on page 53
BD VE support for VXLAN Layer 3 gateway	With L3GW, the device routes VXLAN Layer 2 and Layer 3 traffic over a VXLAN tunnel, and conversely.	VXLAN Layer 3 Gateway on page 79
Logical interfaces	A logical interface (LIF) serves the purpose of abstracting a forwarding interface in a very generic way, making it possible to capture the underlying physical characteristics of a forwarding interface.	Logical Interfaces on page 149
Bridge domains	Bridge domain is an infrastructure that supports the implementation of different switching technologies.	Bridge Domains on page 153
MCT EVPN enhancements and Layer 3 routing for bridge domains	MCT supports the following enhancements: • Auto-generated ESI value for a port channel running LACP • Cluster configuration changes for VLANs and bridge domains that are now configured under an EVPN configuration • Layer 3 routing on an MCT bridge domain	Multi-Chassis Trunking (MCT) on page 111
Pseudowire (PW) control word and flow label	PW control word and PW flow label are mechanisms that improve PW traffic flow over a Multi-Protocol Label Switched packet switched network (MPLS PSN).	Pseudowire control word and flow label on page 168
Routing over VPLS (VEoVPLS)	Support for configuration of a Virtual Ethernet (VE) interface on a Virtual Private LAN Service (VPLS) instance. This enables participation in routing protocols and the exchange of routes with provider edge (PE) devices on remote customer sites.	Routing over VPLS on page 185
MPLS transit load balancing	MPLS transit load balancing provides support for SLX-OS MPLS packet load balancing based on the inner headers when the Layer 2 optimized TCAM profile configuration is activated.	MPLS transit load balancing on page 21
Ability to preserve QoS IP headers in VXLAN packets and in MPLS packets	VXLAN Layer 2 gateways, VXLAN Layer 3 gateways, and VXLAN Layer 2 and Layer 3 gateway interconnections can be configured to support QoS.	QoS for VXLAN Layer 2 gateways on page 77 QoS for VXLAN Layer 3 gateways on page 109 QoS for VXLAN Layer 2 and Layer 3 gateway interconnections on page 107
Multi-chassis trunking	Configuration considerations vary by category: general, peer, VLAN, LSP, and LACP.	MCT configuration considerations on page 122

For more information, see the *Release Notes*.

Link Aggregation

Link aggregation overview

Link aggregation allows you to bundle multiple physical Ethernet links to form a single logical trunk providing enhanced performance and redundancy. The aggregated trunk is referred to as a Link Aggregation Group (LAG). The LAG is viewed as a single link by connected devices, the Spanning Tree Protocol, IEEE 802.1Q VLANs, and so on. When one physical link in the LAG fails, the other links stay up. There is no disruption to traffic.

To configure links to form a LAG, the physical links must be of the same speed. Link aggregation can be done by manually configuring the LAG, or by dynamically configuring the LAG using the IEEE 802.3ad Link Aggregation Control Protocol (LACP).

When queuing traffic from multiple input sources to the same output port, all input sources are given the same weight, regardless of whether the input source is a single physical link or a trunk with multiple member links.

NOTE
The LAG or LAG interface is also referred to as a *port-channel* in the SLX 9850 platform.

The benefits of link aggregation are summarized as follows:

- Increased bandwidth (The logical bandwidth can be dynamically changed as the demand changes.)
- Increased availability
- Load sharing
- Rapid configuration and reconfiguration

Each LAG consists of the following components:

- A MAC address that is different from the MAC addresses of the LAG's individual member links.
- An interface index for each link to identify the link to the neighboring devices.
- An administrative key for each link. Only the links with the same administrative key value can be aggregated into a LAG. On each link configured to use LACP, LACP automatically configures an administrative key value equal to the port-channel identification number.

The SLX 9850 platform supports the following LAG types:

- Static LAG— In static link aggregation, links are added into a LAG without exchanging any control packets between the partner systems. The distribution and collection of frames on static links is determined by the operational status and administrative state of the link.
- Dynamic, standards-based LAG using LACP—Dynamic link aggregation uses LACP to negotiate with links that can be added and removed from a LAG. Typically, two partner systems sharing multiple physical Ethernet links can aggregate a number of those physical links using LACP. LACP creates a LAG on both partner systems and identifies the LAG by the LAG ID. All links with the same administrative key, and all links that are connected to the same partner switch become members of the LAG. LACP continuously exchanges LACPDUs to monitor the health of each member link.

The SLX 9850 platform supports the following trunk type:

- Static, standards-based LAG

The SLX 9850 platform supports the following LAG scalability configuration:

- The **default** profile supports 256 LAGs (64 ports per LAG)
- The **Lag-profile-1** profile supports 512 LAGs (32 ports per LAG)

NOTE

The following example enables the lag hardware profile for scaling to 512. The user has to save and reload to activate a new profile. Use the same procedure to revert to the default profile.

```
device# configure terminal
Entering configuration mode terminal
device(config)# hardware
device(config-hardware)# profile lag lag-profile-1
%Warning: To activate the new profile config, please run 'copy running-config startup-config'
followed by 'reload system'.
device(config-hardware)#
```

LAG load sharing

Extreme devices can be configured for load sharing over a LAG by using the following:

- Hash-based load sharing

Hash-based load sharing

The Extreme device tries to share the traffic load evenly across the ports in LAG group, while ensuring that packets in the flow are not reordered. Individual flows are assigned a LAG index to identify them. An improved hash-based load sharing algorithm has the following enhancements:

- Better distribution
- Support for 32-port LAGs when the maximum number of LAGs in the system is 512.
- Support for 64-port LAGs when the maximum number of LAGs in the system is 256.
- An increased number of fields in the packet header that can be used for load balancing
- Enhanced load sharing in configurations of ECMP with LAGs.

Configuring LAG hashing

To configure symmetric LAG hashing on an SLX 9850 device, complete the following tasks.

1. Define where to start the picking headers for the key generation using the **lag hash hdr-start <fwd |term>** command.
   - fwd— start from the header that is used for the forwarding of the packet (inner header). This is the default option.
   - term— start from the last terminated header (outer header), that is the header below forwarding header. In the case of switching traffic, there would not be any header below forwarding header, thus hashing will not be visible.
2. Configure the number of headers to be considered for LAG hashing using the **lag hash hdr-count <count>** command. The default value is 1. There can be a maximum of 3 headers based on the first header selected using the command in the previous step.

The following options provide other LAG configurations to achieve specific tasks.

- Configure hash rotate using the **lag hash rotate <rotate-number>** command to provide different options for randomness of hashing. The number can be between 0 and 15. The default value is 3.
- Configure hash normalize by using the **lag hash normalize** command if there is a need to use the same hash in both directions. The normalize option is disabled by default.

- Allow the source port to be included in the hashing configuration using the **lag hash srcport** command. The source port is not used for hashing by default.
- To skip the entire MPLS label stack and pick only the BOS label for hashing, use the **lag hash bos <start | skip>**. The command default is If MPLS header is used for hashing, it will use all labels including BOS label for hashing.
  - start— start from BOS. This is the default option.
  - skip— hash from header next to BOS.
- Enter the **lag hash pwctrlword skip** command to skip password control word in the hashing configuration.
- The following MPLS transit node LSR hashing configuration options are available when using the **lag hash speculate-mpls** command. The default option is using the MPLS labels.
  - enable— Enables Speculative MPLS.
  - inner-eth— Enables inner ethernet header hash for L2VPN.
  - inner-ip-raw— Enables inner IPv4 header hash for L2VPN raw mode.
  - inner-ip-tag— Enables inner IPv4 header hash for L2VPN tag mode.
  - inner-ipv6-raw— Enables inner IPv6 header hash for L2VPN raw mode.
  - inner-ipv6-tag— Enables inner IPv6 header hash for L2VPN tag mode.
- Select the fields to be used for LAG hashing per-header type by entering the **[no] lag hash** *protocol-type packet-fields-to-be-used-for-hashing* command.
- Select the protocol header type using one of the following commands. By default, all the header parameters are enabled as shown here. You can disable or enable a parameter only one at any given instant.

  NOTE
  Using the **no** form of the following commands will mask a certain field in the configuration and that field will not be used for load-balance hashing.

  - Ethernet headers. (By default, all the header parameters are enabled as shown here. You can disable or enable a parameter only one at any given instant.) :
    - `[no] load-balance hash ethernet <sa-mac>`
    - `[no] load-balance hash ethernet <da-mac>`
    - `[no] load-balance hash ethernet <etype>`
    - `[no] load-balance hash ethernet <vlan>`
  - IPv4 and L4 headers: `[no] load-balance hash ip < src-ip > <dst-ip > < protocol > < src-l4-port> < dst-l4-port >`
  - IPv6 and L4 headers: `[no] load-balance hash ipv6 < ipv6-src-ip > < ipv6-dst-ip> < ipv6-next-hdr> <ipv6-src-l4-port> < ipv6-dst-l4-port>`
  - MPLS: `[no] load-balance hash mpls < label1 > <label2> < label3>`

Load balancing mechanism on different traffic types

The following table provides information about load balancing on different traffic types.

TABLE 3 Load balancing on different traffic types

Traffic type	Header field	Description
Layer 2/ Layer 3 packet load balancing	• Ethernet DA, SA, Etype, Vlan-id • IPv4/v6 dst IP, src IP • L4 Src-Port, Dst-Port	• Ethernet destination address, source address, ethernet type, VLAN ID load balancing • IPv4/v6 destination address, source address load balancing • Layer 4 source and destination port-based load balancing

TABLE 3 Load balancing on different traffic types (continued)

Traffic type	Header field	Description
VPLS/ VLL packet load balancing	CE to PE router traffic can use the following fields for load-balancing similar to the Layer 2/ Layer 3 traffic) • Ethernet DA, SA, Etype, Vlan-id • IPv4/v6 dst IP, src IP • L4 Src-Port, Dst-Port PE to CE router traffic can use the following fields for load-balancing • Customer (inner) ethernet DA, SA, Etype, Vlan-id • Customer (inner) IPv4/v6 dst IP, Ipv4/Ipv6 src IP, protocol • Customer (inner) L4 Src-Port, Dst-Port	CE to PE router traffic • Ethernet destination address, source address, ethernet type, VLAN ID load balancing • IPv4/v6 destination address, source address load balancing • Layer 4 source and destination port-based load balancing PE to CE router traffic • Customer ethernet destination and source address, ethernet type, VLAN ID load balancing • Customer IPv4/v6 destination address, source address load balancing • Customer Layer 4 source and destination port-based load balancing
MPLS LSR load balancing • Extreme SLX 9850 device provides multiple options to handle different MPLS transit hashing scenarios • The hashing options are mutually exclusive. If one option is enabled, the other option will be disabled.	IP over MPLS traffic going over transit node	Extreme supports speculate-mpls option as default which speculates the IPv4/IPv6 header after the MPLS labels and use the fields for hashing. This hashing scenario is handled by the **lag hash speculate-mpls enable** command in the global mode.
L2VPN (VPLS/VLL) traffic • The hashing options are mutually exclusive. If one option is enabled, the other option will be disabled.	L2VPN tagged mode with IPv4 inner payload	This scenario is handled using the **lag hash speculate-mpls inner-ip-tag** command in the global mode. Some sections of the IPv4 source and destination address fields are also used for load-balance hashing.
	L2VPN raw mode with IPv4 inner payload	This scenario is handled using the **lag hash speculate-mpls inner-ip-raw** command. Some sections of the IPv4 source and destination address fields are also used for load-balance hashing.
	L2VPN tagged mode with IPv6 inner payload	This scenario is handled using the **lag hash speculate-mpls inner-ipv6-tag** command. Some sections of the IPv6 source and destination address fields are also used for load-balance hashing.
	L2VPN raw mode with IPV6 inner payload	This scenario is handled using the **lag hash speculate-mpls inner-ipv6-raw** command. Some sections of the IPv6 source and destination address fields are also used for load-balance hashing.

Displaying LAG hashing

Use the **show port-channel load-balance** command to display the configured parameters for LAG hashing.

```
device# show port-channel load-balance
Header parameters
        Ethernet Mask: sa-mac da-mac etype vlan
        ip: src-ip dst-ip protocol src-l4-port dst-l4-port
        ipv6: ipv6-src-ip ipv6-dst-ip ipv6-next-hdripv6-src-l4-port ipv6-dst-l4-port
        mpls: label1 label2 label3
```

```
Hash Settings
        hdr-start:FWD, hdr-count:1, bos-start:0, bos-skip:0, skip-cw:0
        normalize:0, rotate:3, include_src_port:0, Disable: L2 0, ipv4 0, ipv6 0, mpls 0

mpls_speculate: Enabled

load-balance-type hash-based
```

MPLS transit load balancing

A hashing scheme for enhanced load balancing allows MPLS transit load balancing to include inner headers of different packet types in parallel. With this enhanced load balancing scheme, SLX-OS is capable of load balancing the MPLS packets based on the inner headers like Inner source/destination mac address, Inner IPv4 source/destination address, Inner IPv6 source/destination address, Inner L4 port number if the inner header is IPv4.

This hashing scheme is supported only with the "Layer 2 Optimized" Tcam profile and when the feature is enabled by default in this profile. When the **profile tcam layer2-optimised-1** configuration is activated, the "Error: Operation not supported in the current hardware TCAM profile" message is displayed for the following commands as these functionalities are already taken care of by this enhanced hashing scheme:

- lag hash speculate-mpls inner-eth
- lag hash speculate-mpls inner-ip-raw
- lag hash speculate-mpls inner-ip-tag
- lag hash speculate-mpls inner-ipv6-raw
- lag hash speculate-mpls inner-ipv6-tag

Displaying LAG hashing with "Layer 2 Optimized" Tcam profile

Use the **show port-channel load-balance** command to display the configured parameters for LAG hashing when the **profile tcam layer2-optimised-1** configuration is activated.

```
device# show port-channel load-balance
Header parameters
        Ethernet Mask: sa-mac da-mac etype vlan
        ip: src-ip dst-ip protocol src-l4-port dst-l4-port
        ipv6: ipv6-src-ip ipv6-dst-ip ipv6-next-hdr ipv6-src-l4-port ipv6-dst-l4-port
        mpls: label1 label2 label3

Hash Settings
        hdr-start:FWD, hdr-count:3, bos-start:0, bos-skip:0, skip-cw:0
        normalize:0, rotate:3, include_src_port:0, Disable: L2 0, ipv4 0, ipv6 0, mpls 0

mpls_speculate: Enabled

Inner Header parameters for MPLS packets
               Ethernet Mask: Selective 64 bits from sa-mac da-mac vlan
               ip: src-ip dst-ip src-l4-port dst-l4-port
               ipv6: ipv6-src-ip ipv6-dst-ip

load-balance-type hash-based
```

For comparison, the following displays the **show port-channel load-balance** command output for a non layer 2 optimized profile:

```
device# show port-channel load-balance
Header parameters
        Ethernet Mask: sa-mac da-mac etype vlan
        ip: src-ip dst-ip protocol src-l4-port dst-l4-port
        ipv6: ipv6-src-ip ipv6-dst-ip ipv6-next-hdr ipv6-src-l4-port ipv6-dst-l4-port
        mpls: label1 label2 label3

Hash Settings
```

```
        hdr-start:FWD, hdr-count:1, bos-start:0, bos-skip:0, skip-cw:0
        normalize:0, rotate:3, include_src_port:0, Disable: L2 0, ipv4 0, ipv6 0, mpls 0

mpls_speculate:Enabled

load-balance-type hash-based
```

Link Aggregation Control Protocol

Link Aggregation Control Protocol (LACP) is an IEEE 802.1AX standards-based protocol that allows two partner systems to dynamically negotiate attributes of physical links between them to form logical trunks. LACP determines whether a link can be aggregated into a LAG. If a link can be aggregated into a LAG, LACP puts the link into the LAG. All links in a LAG inherit the same administrative characteristics.

LACP operates in two modes:

- *Active mode*— LACP initiates the LACPDU exchange regardless of whether the partner system sends LACPDUs.
- *Passive mode* — LACP responds to Link Aggregation Control Protocol Data Units (LACPDUs) initiated by its partner system but does not initiate the LACPDU exchange.

LAG distribution process and conditions

The LAG aggregator is associated with the collection and distribution of Ethernet frames. The collection and distribution process is required to guarantee the following:

- Inserting and capturing control PDUs.
- Restricting the traffic of a given conversation to a specific link.
- Load balancing between individual links.
- Handling dynamic changes in LAG membership.

On each port, link aggregation control does the following:

- Maintains configuration information to control port aggregation.
- Exchanges configuration information with other devices to form LAGs.
- Attaches ports to and detaches ports from the aggregator when they join or leave a LAG.
- Enables or disables an aggregator's frame collection and distribution functions.

LAG configuration guidelines:

- Each link in the SLX 9850 hardware can be associated with a LAG; a link cannot be associated with more than one LAG. The process of adding and removing links to and from a LAG is controlled statically or dynamically (through LACP).
- The maximum number of port members that may be assigned to a LAG depends on the LAG profile configuration. By default, the SLX 9850 platform can have a maximum of 256 LAGs with the maximum of 64 ports in each LAG. With the LAG profile 1 , SLX 9850 platform can have a maximum of 512 LAGs with the maximum of 32 ports in each LAG.
- Use the **show hardware profile current** command to view the current LAG profile. When the LAG profile is changed from "default" profile to "LAG-PROFILE-1", the device enables LAG scaling up to 512 LAGs.
- Interfaces configured as switchport interfaces cannot be aggregated into a LAG. However, a LAG can be configured as a switchport.

Configuring and managing Link Aggregation

The following sections discuss working with Link Aggregation on Extreme devices.

Configuring a new port channel interface

Follow this procedure to create a new port channel interface at the global configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface port-channel** command to create a new port channel interface at the global configuration level.

```
device(config)# interface port-channel 30
```

NOTE
The port-channel interface ranges from 1 to 512.

The following example creates a new port channel interface of 30.

```
device# configure terminal
device(config)# interface port-channel 30
```

After creating a new port channel, you can do "no shutdown" or "shutdown" to bring up or down the port-channel as follows.

```
device# configuration terminal
device(config)# interface Port-channel 30
2016/10/17-20:31:21, [NSM-1004], 302, M2 | Active | DCE, INFO, SLX,  Port-channel 30 is created.
device(config-Port-channel-30)#
device(config-Port-channel-30)# no shutdown
2016/10/17-20:31:26, [NSM-1019], 303, M2 | Active | DCE, INFO, SLX,  Interface Port-channel 30 is
administratively up.
device(config-Port-channel-30)#
```

Deleting a port channel interface

Follow this procedure to delete a port channel interface and all member interfaces from the specified LAG at the global configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **no interface port-channel** command to delete an existing port channel interface at the global configuration level.

```
device(config)# no interface port-channel 30
```

NOTE
The port-channel interface ranges from 1 to 512.

The following example deletes the existing port channel interface 30.

```
device# configure terminal
device(config)# no interface port-channel 30
```

Adding a member port to a port channel

Follow this procedure to add a port to a specific port channel interface at the interface configuration level. If the port channel is not created, the **channel-group** command creates the port channel and also adds a port to the port channel.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **interface port-channel** command to add a port channel interface at the global configuration level.

```
device(config)# interface port-channel 30
device(conf-Port-channel-30)#
```

3. Configure the **interface ethernet** command to enable the interface.

```
device(conf-Port-channel-30)# interface ethernet 1/5
device(conf-if-eth-1/5)#
```

4. Add a port to the port channel interface as static.

```
device(conf-if-eth-1/5)# channel-group 30 mode on
```

5. Add a port to the port channel interface as a dynamic (using LACP), active or passive mode.

```
device(conf-if-eth-1/5)# channel-group 30 mode active

device(conf-if-eth-1/5)# channel-group 30 mode passive
```

The following example is for a static LAG configuration with the mode ON.

```
device# configure terminal
device(config)# interface port-channel 30
device(conf-Port-channel-30)# interface ethernet 1/5
device(conf-if-eth-1/5)# channel-group 30 mode on
```

The following example adds a port 1/5 to the existing dynamic port channel interface 30 with the mode active.

```
device# configure terminal
device(config)# interface port-channel 30
device(conf-Port-channel-30)# interface ethernet 1/5
device(conf-if-eth-1/5)# channel-group 30 mode active
```

NOTE

Run the **no shutdown** command to bring the above interface online.

```
device(conf-if-eth-1/5)# no shutdown
2016/10/18-03:47:15, [NSM-1019], 528, M2 | Active | DCE, INFO, SLX,  Interface Ethernet 1/5 is
administratively up.2016/10/18-03:47:15, [NSM-1001], 529, M2 | Active | DCE, INFO, SLX,  Interface
Ethernet 1/5 is online.
```

The following example adds a port 1/5 to the existing dynamic port channel interface 30 with the mode passive.

```
device# configure terminal
device(config)# interface port-channel 30
device(conf-Port-channel-30)# interface ethernet 1/5
device(conf-if-eth-1/5)# channel-group 30 mode passive
```

Deleting a member port from a port channel

Follow this procedure to delete a member port from a port channel interface at the interface configuration level.

Delete a port from the port channel interface.

```
device(conf-if-eth-1/5)# no channel-group
```

The following example deletes a port 1/5 from the existing port channel interface 30.

```
device# configure terminal
device(config)# interface ethernet 1/5
device(conf-if-eth-1/5)# no channel-group
```

Configuring the minimum number of LAG member links

Follow this procedure to configure the minimum number of LAG member links that should be functional so that the port-channel interface is operationally up.

This configuration allows a port-channel to operate at a certain minimum bandwidth at all times. If the bandwidth of the port-channel drops below the minimum number, then the port-channel is declared operationally DOWN even though it has operationally UP members.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **interface port-channel** command at the global configuration level.

```
device(config)# interface port-channel 30
device(conf-Port-channel-30)#
```

3. Configure the minimum number of LAG member links at the port-channel interface configuration mode.

```
device(conf-Port-channel-30)# minimum-links 5
```

NOTE
The number of links ranges from 1 to 64. The default minimum links is 1.

The following example sets min-link 5 to the existing port channel interface 30.

```
device# configure terminal
device(config)# interface port-channel 30
device(conf-Port-channel-30)# minimum-links 5
```

Configuring the LACP system priority

You configure the LACP system priority on each switch running LACP. LACP uses the system priority with the switch MAC address to form the system ID and also during negotiation with other switches.

The system priority value must be a number in the range of 1 through 65535. The higher the number, the lower the priority. The default priority is 32768.

To configure the global LACP system priority, perform the following steps:

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Specify the LACP system priority.

```
device(config)# lacp system-priority 25000
```

3. To reset the system priority to the default value.

```
device(config)# no lacp system-priority
```

Configuring the LACP port priority

Follow this procedure to configure the LACP port priority of a member port of a specific port-channel interface.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **interface port-channel** command to add a port channel interface at the global configuration level.

```
device(config)# interface port-channel 30
device(conf-Port-channel-30)#
```

3. Configure the **interface ethernet** command and add the port to the port-channel interface.

```
device(conf-Port-channel-30)# interface ethernet 1/5
device(conf-if-eth-1/5)#channel-group 30 mode active
```

4. Configure the LACP port priority 12 for the member port.

```
device(conf-if-eth-1/5)# lacp port-priority 12
```

NOTE

The LACP port priority value ranges from 1 to 65535. The default value is 32768.

5. To rest the configured port priority to the default value.

```
device(conf-if-eth-1/5)# no lacp port-priority
```

The example sets the port priority as 12.

```
device# configure terminal
device(config)# interface port-channel 30
device(conf-Port-channel-30)# interface ethernet 1/5
device(conf-if-eth-1/5)# channel-group 30 mode active
device(conf-if-eth-1/5)# lacp port-priority 12
```

Configuring the LACP timeout period

The LACP timeout period indicates how long LACP waits before timing out the neighboring device. The **short** timeout period is 3 seconds and the **long** timeout period is 90 seconds. The default is **long**. The **short** timeout period specifies that the PDU is sent every second and the port waits three times this long (three seconds) before invalidating the information received earlier on this PDU. The **long** timeout period specifies that the PDU is sent once in 30 seconds and the port waits three times this long (90 seconds) before invalidating the information received earlier on this PDU.

To configure the LACP timeout period on an interface, perform the following steps:

1. Enter the **configure terminal** command to access global configuration mode.
2. Enter the **interface** command, specifying the interface type and the slot/port.

```
device(config)# interface ethernet 1/1
```

3. Enter the **no shutdown** command to enable the interface.
4. Specify the LACP timeout short period for the interface.

```
device(conf-if-eth 1/1)# lacp timeout short
```

5. Specify the LACP timeout long period for the interface.

```
device(conf-if-eth 1/1)# lacp timeout long
```

Configuring LACP default Up

Follow this procedure to activate an LACP link even in the absence of PDUs.

Consider the following when using the **lacp default-up** command:

- The command is available only if the configured interface is a dynamic member of a port-channel interface.
- The command is not supported on static LAGs.
- The command is not supported on port-channel interfaces.

1. Enter the **configure terminal** command to access global configuration mode.
2. Enter the **interface** command, specifying the interface type and the slot/port.

```
device(config)# interface ethernet 1/1
```

3. Specify LACP default-up for the interface.

```
device(conf-if-eth-1/1)# lacp default-up
```

4. Enter the no form of the command to disable the configuration.

```
device(conf-if-eth-1/1)#  no lacp default-up
```

LACP PDU forwarding

By default, LACP PDUs received on an interface where LACP is not configured are discarded. For scenarios in which the interface requires LACP PDU packet forwarding, you can configure the device to forward the LACP PDU on the VLAN on which it is received using the **lacp-pdu-forward enable** command in the interface configuration mode or port channel configuration mode.

Since the destination address of the PDU is a multicast MAC, the frame will be flooded on the VLAN. If the VLAN on which the LACP PDU is received is a regular VLAN, the PDU will be flooded on the VLAN. If the VLAN on which the PDU is received is a service delimiter for a bridge domain, the LACP PDU is flooded on the bridge domain accordingly.

LACP PDU forwarding is supported only on physical interfaces and static port channel interfaces. LACP PDUs cannot be forwarded if they are received on a LACP based dynamic port channel. LACP PDU forwarding enabled on a static port channel applies to all the member ports. If LACP is enabled on a port, it overrides the LACP PDU forwarding configuration and the PDUs are trapped in the CPU.

Configuring LACP PDU forwarding on a port-channel interface

Perform the following steps to configure LACP PDU forwarding on a port-channel interface.

1. Enter the global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **interface port-channel** command to add a port channel interface at the global configuration level.

```
device(config)# interface port-channel 10
device(conf-Port-channel-10)#
```

3. Configure LACP PDU forwarding on the port-channel interface.

```
device(conf-Port-channel-10)# lacp-pdu-forward enable
```

LACP PDU forwarding is supported only on static port channel interfaces.

The following example enables LACP forwarding on a port-channel interface.

```
device# configure terminal
device(config)# interface port-channel 10
device(conf-Port-channel-10)# lacp-pdu-forward enable
```

Configuring LACP PDU forwarding on a physical interface

Perform the following steps to configure LACP PDU forwarding on a physical interface.

1. Enter the global configuration mode.

```
device# configure terminal
device(config)#
```

2. Specify the physical interface on which LACP PDU forwarding needs to enabled.

```
device(config)# interface ethernet 4/1
device(conf-if-eth-4/1)#
```

3. Configure LACP PDU forwarding on the physical interface.

```
device(conf-if-eth-4/1)# lacp-pdu-forward enable
```

The following example enables LACP forwarding on a port-channel interface.

```
device# configure terminal
device(config)# interface ethernet 4/1
device(conf-if-eth-4/1)# lacp-pdu-forward enable
```

Displaying port-channel information

Various show commands are used to display information for a specific port-channel interface.

Before displaying the port-channel information, you should have created a port-channel interface in a LAG to generate details.

1. Use the **show port-channel summary** command to display brief information of all port-channels.

```
device# show port-channel summary
Flags:  D - Down                P - Up in port-channel (members)
        U - Up (port-channel)   * - Primary link in port-channel
        S - Switched
        M - Not in use. Min-links not met
===== =============== ========== ===============
Group Port-channel    Protocol   Member ports
===== =============== ========== ===============
1     Po 1    (D)     None       Eth 2/125 (D)
                                 Eth 4/125 (D)
2     Po 2    (D)     None       Eth 2/126 (D)
                                 Eth 4/126 (D)
10    Po 10   (U)     LACP       Eth 2/4* (P)
                                 Eth 2/18 (P)
100   Po 100  (U)     None       Eth 2/10* (P)
                                          Eth 2/11 (P)
```

2. Use the **show port-channel detail** command to display detailed information of all the port-channels.

```
device# show port-channel detail
 Static Aggregator: Po 1
 Aggregator type: Standard
 Number of Ports: 2
 Member ports:
   Eth 2/125
   Eth 4/125


 Static Aggregator: Po 2
 Aggregator type: Standard
 Number of Ports: 2
 Member ports:
   Eth 2/126
   Eth 4/126


 Static Aggregator: Po 100
 Aggregator type: Standard
 Number of Ports: 2
 Member ports:
   Eth 2/10   *
   Eth 2/11


 LACP Aggregator: Po 10
Aggregator type: Standard
  Actor System ID - 0x8000,76-8e-f8-0a-98-00
  Admin Key: 0010 - Oper Key 0010
  Receive link count: 2 - Transmit link count: 2
  Individual: 0 - Ready: 1
  Partner System ID - 0x8000,76-8e-f8-0a-68-00
  Partner Oper Key 0010
 Number of Ports: 2
 Member ports:
   Link: Eth 2/4 (0x18820016) sync: 1   *
   Link: Eth 2/18 (0x18890084) sync: 1
```

3. Use the **show port-channel** *number* command to display detailed information of a specific port-channel interface

```
device# show port-channel 10
Port-channel 10 is admin down, line protocol is down (admin down)
Hardware is AGGREGATE, address is 00e0.0c70.cc07
    Current address is 00e0.0c70.cc07
Interface index (ifindex) is 671088650
Minimum number of links to bring Port-channel up is 1
MTU 1548 bytes
LineSpeed Actual     : Nil
Allowed Member Speed : 10000 Mbit
Priority Tag disable
Forward LACP PDU: Enable
Last clearing of show interface counters: 00:29:09
Queueing strategy: fifo
Receive Statistics:
    0 packets, 0 bytes
    Unicasts: 0, Multicasts: 0, Broadcasts: 0
    64-byte pkts: 0, Over 64-byte pkts: 0, Over 127-byte pkts: 0
    Over 255-byte pkts: 0, Over 511-byte pkts: 0, Over 1023-byte pkts: 0
    Over 1518-byte pkts(Jumbo): 0
    Runts: 0, Jabbers: 0, CRC: 0, Overruns: 0
    Errors: 0, Discards: 0
Transmit Statistics:
    0 packets, 0 bytes
    Unicasts: 0, Multicasts: 0, Broadcasts: 0
    Underruns: 0
    Errors: 0, Discards: 0
Rate info:
    Input 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
    Output 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
Time since last interface status change: 00:29:09
```

Displaying LACP system-id information

Follow this procedure to display LACP system ID and priority information.

Enter the **show lacp sys-id** command to display LACP information for the system ID and priority.

```
device# show lacp sys-id
System ID: 0x8000,76-8e-f8-0a-98-00
```

Displaying LACP statistics

Follow this procedure to display LACP statistics for a port-channel interface or for all port-channel interfaces.

Before displaying the LACP port-channel information, it is recommended that you create a port-channel interface in a LAG to generate details.

Enter the **show lacp counters** command to display LACP statistics for a port-channel.

```
device# show lacp counter
Traffic statistics
Port            LACPDUs            Marker           Pckt err     Sent     Recv     Sent     Recv
Sent     Recv
Aggregator Po 3  Eth 1/6                          110      0           0        0
0        0
```

Clearing LACP counter statistics on a LAG

This topic describes how to clear LACP counter statistics on a single LAG.

To clear LACP counter statistics on a LAG, use the following command:

Enter the **clear lacp** *LAG_group_number* **counters** command to clear the LACP counter statistics for the specified LAG group number.

```
device# clear lacp 42 counters
```

Clearing LACP counter statistics on all LAG groups

This topic describes how to clear the LACP counter statistics for all LAG groups.

To clear LACP counter statistics on all LAG groups, use the following command:

Enter the **clear lacp counter** command to clear the LACP counter statistics for all LAG groups.

```
device# clear lacp counter
```

Troubleshooting LACP

To troubleshoot problems with your LACP configuration, use the following troubleshooting tips.

If a standard IEEE 802.1AX-based dynamic trunk is configured on a link and the link is not able to join the LAG, do the following:

- Make sure that both ends of the link are configured as **standard** for the trunk type.
- Make sure that both ends of the link are *not* configured for **passive** mode. They must be configured as **active** /**active**, **active** / **passive**, or **passive** /**active**.
- Make sure that the port-channel interface is in the administrative "up" state by ensuring that the **no shutdown** command was entered on the interface on both ends of the link.
- Make sure that the links that are part of the LAG are connected to the same neighboring switch.
- Make sure that the system ID of the switches connected by the link is unique. You can verify this by entering the **show lacp sys-id** command on both switches.
- Make sure that LACPDUs are being received and transmitted on both ends of the link and that there are no error PDUs. You can verify this by entering the **show lacp counters** *number* command and looking at the receive mode (rx) and transmit mode (tx) statistics. The statistics should be incrementing and should not be at zero or a fixed value. If the PDU rx count is not incrementing, check the interface for possible CRC errors by entering the **show interface** *link-name* command on the neighboring switch. If the PDU tx count is not incrementing, check the operational status of the link by entering the **show interface** *link-name* command and verifying that the interface status is "up."

When a link has problem, the **show port-channel** command displays the following message:

```
Mux machine state: Deskew not OK.
```

If a static trunk is configured on a link and the link is not able to join the LAG, do the following:

- Make sure that both ends of the link are configured as **standard** for trunk type and verify that the mode is "on."
- Make sure that the port-channel interface is in the administrative "up" state by ensuring that the **no shutdown** command was entered on the interface on both ends of the link.

Unidirectional Link Detection

Unidirectional Link Detection overview

Unidirectional Link Detection (UDLD) monitors a link between two Extreme devices and blocks the ports on both ends of the link if there is a unidirectional failure.

UDLD protocol detects and blocks broken unidirectional links in the network. This is done through the exchange of UDLD protocol data units (PDU) between devices on a physical link. Both ends of the link must support the same proprietary UDLD protocol to detect the unidirectional link condition.

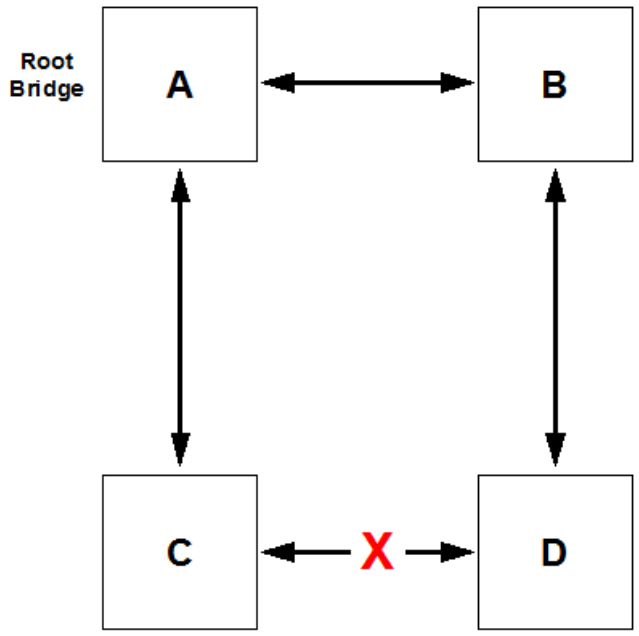
A unidirectional link is assumed when the UDLD stops receiving UDLD PDUs from the other end of the link. The device then blocks the physical link. The physical link will still be up but the line protocol will be down. UDLD PDUs continue to be transmitted and received on the link.

UDLD is disabled by default. To use the UDLD protocol, the protocol must first be enabled globally and then on each individual physical port. When enabled globally and on a physical port, the device starts transmitting UDLD PDUs periodically on the port.

How UDLD works

The following shows a simple four-switch network in which two paths connect to each switch. STP blocks traffic on as many ports as necessary so that only one operational path exists from the STP root bridge to all nodes in the network.

FIGURE 1 Four-switch example for UDLD



In the previous figure, STP detects that the port on Switch D that is connected to Switch C should be put into a blocked state. Therefore, no data traffic gets transmitted or received on this port. Data traffic remains blocked as long as Switch D receives bridge protocol data units (BPDUs) from both switches C and B.

If the link between Switch C and Switch D becomes unidirectional (for reasons such as hardware failure or incorrect cabling) in the direction from D to C, Switch D ages out the status that it was receiving BPDUs from Switch C. This eventually causes STP to put the port in a forwarding state, thus allowing all data traffic. This creates a loop for all BUM traffic that enters the network. BUM traffic can go from Switch B to Switch D to Switch C to Switch A, and then back to Switch B.

To prevent this loop from forming, UDLD can be used to detect that the link between Switch C and Switch D has become unidirectional.

UDLD considerations and restrictions

Note the following for UDLD:

- UDLD is used in conjunction with the Spanning Tree Protocol.
- UDLD runs only on physical ports assigned to a port channel.
- UDLD is supported on directly connected switches only.
- The protocol must be running on both ends of the link.
- The default timeout is 2.5 (2500 ms) seconds.
- Tagged UDLD is not supported.
- The feature is not compatible with Cisco UDLD protocol.
- Extreme uses a proprietary implementation that interoperates with Multi-Service IronWare devices, FastIron devices, and VDX devices.
- The UDLD interface statistics lose some accuracy after a failover.
- Upon executing the **no protocol udld** command, all of the UDLD global and UDLD interface configuration changes will be removed and the protocol will revert back to its initial disabled state. This means that all interfaces that have been enabled for the protocol stops transmitting and receiving UDLD PDUs.

Configuring UDLD

Complete the following steps to configure basic UDLD on your device.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable the UDLD protocol and enter protocol UDLD configuration mode.

   ```
   device(config)# protocol udld
   ```

3. Change the hello interval to 2000 milliseconds.

   ```
   device(config-udld)# hello 20
   ```

   This changes the interval at which UDLD PDUs are transmitted. The default interval, in counts of one hundred milliseconds is 500 ms.

4. Change the timeout multiplier from the default of 5.

```
device(config-udld)# multiplier 8
```

   This changes the timeout multiplier value to affect the UDLD PDU timeout interval. The UDLD timeout interval is the product of the hello time interval at the other end of the link and the timeout multiplier value.

   When the remote is Multi-Service IronWare or FastIron devices, the timeout equals local hello interval × multiplier value.

5. Return to global configuration mode.

```
device(config-udld)# exit
```

6. Enter interface configuration mode for an port.

```
device(config)# interface ethernet 5/1
```

7. Enable UDLD on the interface.

```
device(config-if-eth-5/1)# udld enable
```

8. Repeat the preceding step for each port on which you wish to enable UDLD.

   NOTE

   When the UDLD protocol is enabled on one end of a link, the timeout period might elapse before the UDLD protocol is enabled on the other end of the link. In this case, the link becomes temporarily blocked. When the UDLD protocol is enabled at the other end of the link and a UDLD PDU is received, UDLD automatically unblocks the link.

9. Return to privileged exec mode.

```
device(config-if-eth-5/1)# end
```

10. Verify the configuration.
    - Show the UDLD global configuration.

```
device# show udld
UDLD Global Information
  Admin State:       UDLD enabled
  UDLD hello time:   1000 milliseconds
  UDLD timeout:      5000 milliseconds
```

    - Show UDLD status for an interface

```
device# show udld interface ethernet 2/6
Global Admin State:  UDLD enabled

UDLD information for Ethernet 2/6
  UDLD Admin State:                  Enabled
  Interface Operational State:       Bidirectional link
  Remote hello time:                 Unknown
  Remote MAC Addr:                   0024.3890.0d81
  Local system id:  0x9f01fee0       Remote system id: 0x24900c00
  Local port :      2/6              Remote port :     9/2
  Local link id:    0x0              Remote link id:   0x0
  Last Xmt Seq Num: 43849            Last Rcv Seq Num: 43880
```

    - Show UDLD statistics.

```
device# show udld statistics
UDLD Interface statistics for Ethernet 2/7
Frames transmitted:         260
Frames received:            223
Frames discarded:           0
Frames with error:          0
Remote port id changed:     0
Remote MAC address changed: 0
```

      NOTE

      The **show interface** command also indicates whether UDLD is enabled.

11. Save the configuration.

```
device# copy running-config startup-config
```

UDLD configuration example

```
device# configure terminal
device(config)# protocol udld
device(config-udld)# hello 20
device(config-udld)# multiplier 8
device(config-udld)# exit
device(config)# interface ethernet 5/1
device(config-if-eth-5/1)# udld enable
device(config-if-eth-5/1)# end
device# show udld
device# copy running-config startup-config
```

VLANs

802.1Q VLAN overview

IEEE 802.1Q VLANs provide the capability to overlay the physical network with multiple virtual networks. VLANs allow you to isolate network traffic between virtual networks and reduce the size of administrative and broadcast domains.

A VLAN contains end stations that have a common set of requirements that are independent of physical location. You can group end stations in a VLAN even if they are not physically located in the same LAN segment. VLANs are typically associated with IP subnetworks and all the end stations in a particular IP subnet belong to the same VLAN. Traffic between VLANs must be routed. VLAN membership is configurable on a per-interface basis.

Configuring VLANs

The following sections discuss working with VLANs on Extreme devices.

Configuring a VLAN

Follow this procedure to configure a VLAN in the Extreme device at the global configuration level.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **vlan** command to create a topology group at the global configuration level.

```
device(config)# vlan 5
device(config-vlan-5)#
```

NOTE

The **no vlan** command removes the existing VLAN instance from the device.

Configuring a switchport interface

Follow this procedure to configure a switchport interface in the device to send and receive data packets.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to configure the interface mode.

```
device(config)# interface ethernet 0/1
```

3. Enter the **switchport** command to configure a switchport interface.

```
device(conf-if-eth-0/1)# switchport
```

Configuring the switchport interface mode

Do the following to set the switchport interface as access or trunk. This configuration works only when the interface is set as switchport.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to configure the interface mode.

```
device(config)# interface ethernet 0/1
```

3. Enter the **switchport** command to set the interface as switchport.

```
device(conf-if-eth-0/1)# switchport
```

4. Enter the **switchport mode** command to configure the switchport interface in trunk mode.

```
device(conf-if-eth-0/1)# switchport mode trunk
```

NOTE
The default mode is access. Enter the **switchport mode access** command to set the mode as *access*.

NOTE
Before you change the switch port mode from **switchport mode access** with an explicit **switchport access vlan** to **switchport mode trunk-no-default-native**, you must enter the **no switchport** command on the interface level, and then enter the **switchport** command to set the interface as a switchport. Now you can configure the **switchport mode trunk-no-default-native** command.

Configuring the switchport access VLAN type

Do the following to change the switchport access VLAN type. This configuration works only when the interface is set as switchport.

Ensure that reserved VLANs are not used. Use the **no switchport access vlan** command to set the default VLAN as the access VLAN.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to specify an Ethernet interface.

```
device(config)# interface ethernet 0/1
```

3. Enter the **switchport** command to set the interface as switchport.

```
device(conf-if-eth-0/1)# switchport
```

4. Enter the **switchport access vlan** command to set the mode of the interface to *access* and specify a VLAN.

```
device(conf-if-eth-0/1)# switchport access vlan 10
```

This example sets the mode of a specific port-channel interface to *trunk*.

```
device# configure terminal
device(config)# interface port-channel 35
device(config-port-channel-35)# switchport mode trunk
```

Configuring a VLAN in trunk mode

Do the following to add or remove VLANs on a Layer 2 interface in trunk mode. The configuration is also used to configure the VLANs to send and receive data packets.

Ensure that reserved VLANs are not used.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to specify an Ethernet interface.

```
device(config)# interface ethernet 0/1
```

3. Enter the **switchport** command to set the interface as switchport.

```
device(conf-if-eth-0/1)# switchport
```

4. Enter the **switchport trunk allowed vlan** command to set the mode of the interface to *trunk* and add a VLAN.

```
device(conf-if-eth-0/1)# switchport trunk allowed vlan add 5
```

The example sets the mode of the Ethernet interface to *trunk*.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# switchport mode trunk
```

The example sets the mode of a port-channel interface to *trunk* and allows all VLANs.

```
device# configure terminal
device(config)# interface port-channel 35
device(config-Port-channel-35)# switchport trunk allowed vlan all
```

Configuring a native VLAN on a trunk port

Do the following to set native VLAN characteristics on a trunk port for classifying the untagged traffic data packets.

Ensure that reserved VLANs are not used.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **interface ethernet** command to configure the interface mode.

```
device(config)# interface ethernet 0/1
```

3. Enter the **switchport** command to set the interface as switchport.

```
device(conf-if-eth-0/1)# switchport
```

4. Enter the **switchport trunk native-vlan** command to set native VLAN characteristics to *access* and specify a VLAN.

```
device(conf-if-eth-0/1)# switchport trunk native-vlan 300
```

This example removes the configured native VLAN on the Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# no switchport trunk native-vlan 300
```

Enabling VLAN tagging for native traffic

Do the following to enable tagging for native traffic on a specific interface.

Ensure that reserved VLANs are not used.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to configure the interface mode.

```
device(config)# interface ethernet 0/1
```

3. Enter the **switchport** command to set the interface as switchport.

```
device(conf-if-eth-0/1)# switchport
```

4. Enter the **switchport trunk tag native-vlan** command to enable tagging for native traffic data VLAN characteristics on a specific interface.

```
device(conf-if-eth-0/1)# switchport trunk tag native-vlan
```

This example enables tagging for native traffic data on a specific Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# switchport trunk tag native-vlan
```

This example disables the native VLAN tagging on a port-channel.

```
device# configure terminal
device(config)# interface port-channel 35
device(config-Port-channel-35)# no switchport trunk tag native
```

Displaying the status of a switchport interface

Do the following to display detailed Layer 2 information for all switchport interfaces.

Enter the **show interface switchport** to display the detailed Layer 2 information for all interfaces.

```
device# show interface switchport
 Interface name          : Eth 0/1
 Switchport mode         : access
 Ingress filter          : enable
 Acceptable frame types : all
 Default Vlan            : 1
 Active Vlans            : 1
 Inactive Vlans          : -
 Interface name          : Port-channel 5
 Switchport mode         : access
 Ingress filter          : enable
 Acceptable frame types : all
 Default Vlan            : 1
 Active Vlans            : 1
```

Displaying the switchport interface type

Do the following to display detailed Layer 2 information for a specific interface.

Enter the **show interface switchport** to display the detailed Layer 2 information for a specific interface.

```
device# show interface ethernet 0/1 switchport
 Interface name           : ethernet 0/1
 Switchport mode          : trunk
 Fcoeport enabled         : no
 Ingress filter           : enable
 Acceptable frame types   : vlan-tagged only
 Native Vlan              : 1
 Active Vlans             : 1,5-10
 Inactive Vlans           : -
```

The example displays the detailed Layer 2 information for a port-channel interface.

```
device# show interface port-channel 5 switchport
 Interface name           : Port-channel 5
 Switchport mode          : access
 Fcoeport enabled         : no
 Ingress filter           : enable
 Acceptable frame types   : vlan-untagged only
 Default Vlan             : 1
 Active Vlans             : 1
 Inactive Vlans           : -
```

Verifying a switchport interface running configuration

Do the following to display the running configuration information for the Layer 2 properties for a specific interface.

Enter the **show running-config interface** to display the running configuration information for a specific interface.

```
device# show running-config interface ethernet 0/1 switchport
interface interface Eth 0/1
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 5-10
 switchport trunk tag native-vlan
```

This example displays the running configuration information for a port-channel interface.

```
device# show running-config interface port-channel 5 switchport
interface Port-channel 5
 switchport
 switchport mode access
 switchport access vlan 1
```

Displaying VLAN information

Do the following to display information about a specific VLAN.

Enter the **show vlan** to display information about VLAN 1.

```
device# show vlan 1
VLAN Name State Ports
(u)-Untagged, (t)-Tagged
(c)-Converged
================================================================
1 default ACTIVE Eth 0/1(t) Eth 0/4(t) Eth 0/5(t) Eth 0/8(t)
```

Enabling Layer 3 routing for VLANs

Do the following to enable Layer 3 routing on a VLAN.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a VLAN.

```
device(config)# vlan 200
```

3. Create a virtual Ethernet (VE), assign an IP address and mask, and enable the interface.

```
device(config)# interface ve 200
device(config-Ve-200)# ip address 10.2.2.1/24
device(config-Ve-200)# no shutdown
```

A VE interface can exist without a VLAN configuration, but it must be provisioned in the VLAN in order to be used.

4. Enter the **router-interface** command and specify the VLAN.

```
device(config-vlan-200)# router-interface ve 200
```

VLAN statistics

Devices gather statistics for all ports and port channels on configured VLANs.

Use the **statistics** command in the VLAN configuration mode to enable statistics on a VLAN.

NOTE

Statistics has to be manually enabled for a specific VLAN, since it is not enabled by default for VLANs.

Please note that:

- The statistics reported are not real-time statistics since they depend upon the load on the system.
- Statistics has to be manually enabled for a specific VLAN. This ensures better utilization of the statistics resources in the hardware.
- Statistics for VLANs with VE interfaces consider only the switched frames. Packets which are routed into or out of the VE interface are not counted.
- Enabling statistics on a VLAN has a heavy impact on the data traffic.

Enabling statistics on a VLAN

Follow this procedure to enable statistics on a VLAN.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Enter the **vlan** command to specify a VLAN for statistics collection.

```
device(config)# vlan 5
device(config-vlan-5)#
```

3. Enter the **statistics** command to enable statistics for all ports and port channels on configured VLANs.

```
device(config-vlan-5)# statistics
```

NOTE

Use the **no statistics** command to disable statistics on VLANs.

```
device(config-vlan-5)# no statistics
```

Displaying statistics for VLANs

Do the following to display statistics information for VLANs.

Enter the **show statistics vlan** command to view the statistics for all ports and port channels on all configured VLANs.

```
device# show statistics vlan

Vlan 10 Statistics
Interface     RxPkts          RxBytes          TxPkts          TxBytes
eth 0/1       821729          821729           95940360        95940360
eth 0/2       884484          885855           95969584        95484555
po  1        8884            8855              9684            9955

Vlan 20 Statistics
Interface     RxPkts          RxBytes          TxPkts          TxBytes
eth 0/6       821729          821729           95940360        95940360
eth 0/21     8884            8855              9684            9955
po  2      884484          885855           95969584        95484555
```

TABLE 4 Output descriptions of the show statistics vlan command

Field	Description
Interface	The interface whose counter statistics are displayed.
RxPkts	The number of packets received at the specified port.
RxBytes	The number of bytes received at the specified port.

TABLE 4 Output descriptions of the show statistics vlan command (continued)

Field	Description
TxPkts	The number of packets transmitted from the specified port.
TxBytes	The number of bytes transmitted from the specified port.

Displaying VLAN statistics for a specific VLAN

Enter the **show statistics vlan** *vlan ID* command to view the statistics for a specific VLAN. Here *vlan ID* is the specific VLAN ID.

```
device# show statistics vlan 10

Vlan 10 Statistics
Interface     RxPkts          RxBytes         TxPkts          TxBytes
eth 0/1       821729          821729          95940360        95940360
eth 0/2       884484          885855          95969584        95484555
po  1      8884            8855            9684            9955
```

Clearing statistics on VLANs

Follow the procedure to clear statistics' information for VLANs.

Enter the **clear statistics vlan** command to clear the statistics for all ports and port channels on all configured VLANs.

```
device# clear statistics vlan
```

Clearing statistics for a specific VLAN

Enter the **clear statistics vlan** *vlan ID* command to clear the statistics for a specific VLAN. Here *vlan ID* is the specific VLAN ID.

```
device# clear statistics vlan 10
```

VE route-only mode

By default, physical ports and port-channels (LAG ports) support both Layer 2 switching and Layer 3 routing. VE route-only mode enables these ports to act exclusively as a Layer 3 virtual Ethernet (VE) interface, dropping all ingress packets that require Layer 2 switching.

The MAC learning of dropped packets is not affected by this feature. ARP requests (broadcast), LACP, and BPDU packet processing are also not affected. The following table lists the effects of this mode on a variety of features.

TABLE 5 Effects of VE route-only mode on features

Feature	Ingress port as route-only port (incoming frames)	Egress port as route-only port (outgoing frames)
Packets requiring switching	Drop, learn MAC address	Forwarded/switched
Packets requiring routing	Forwarded	Forwarded
ARP requests	Trapped/punted, ARP response generated	Forwarded
LACP packets	Trapped/punted and processed	Forwarded
STP/BPDU packets	Trapped/punted and processed	Forwarded

Note the following considerations and limitations:

- Egress packets through a port configured as route-only are transmitted irrespective of whether they are switched or routed.

- This feature is enabled on the active management module (MM), and is available on the other MM after a failover.
- The number of TCAM entries required for this feature depends on the maximum number of physical ports. On a there are approximately 40 physical ports per PPE , and a maximum of 512 LAG ports for the entire system. Therefore the maximum number of TCAM entries for route-only support is 40. These entries are available on all TCAM profiles.

Configuring VE route-only mode on a physical port

Do the following to configure VE route-only mode on a physical port.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a VLAN.

```
device(config)# vlan 100
```

3. Specify an Ethernet interface.

```
device(config)# interface ethernet 1/2
```

4. Enter the **switchport** command to configure Layer 2 characteristics.

```
device(conf-if-eth-1/2)# switchport
```

5. Specify trunk mode.

```
device(conf-if-eth-1/2)# switchport mode trunk
```

6. Tag the port to a VLAN.

```
device(conf-if-eth-1/2)# switchport mode trunk allowed vlan add 100
```

7. Enter the **route-only** command to enable Layer 3 routing exclusively on the port.

```
device(conf-if-eth-1/2)# route-only
```

   Use the **no route-only** command to revert to default Layer 2 and Layer 3 behavior.

8. Enable the interface and exit to global configuration mode.

```
device(conf-if-eth-1/2)# no shutdown
device(conf-if-eth-1/2)# exit
```

9. Verify the Ethernet configuration.

```
device(conf-if-eth-1/2)# do show running-cocnfig interface ethernet 1/2
switchport
switchport mode trunk
switchport trunk allowed vlan add 100
route only
no shutdown
```

10. Verify the port statistics for switching packets dropped.

```
device(conf-if-eth-1/2)# do show interface ethernet 1/2
               Ethernet 1/2 is up, line protocol is up (connected)
               Hardware is Ethernet, address is 768e.f80a.033c
               Current address is 768e.f80a.033c
          ….
         Rate info:
               Input 0.001008 Mbits/sec, 0 packets/sec, 0.00% of line-rate
               Output 0.000252 Mbits/sec, 0 packets/sec, 0.00% of line-rate
               Route-Only Packets Dropped: 17
               Time since last interface status change: 21:35:41
```

11. Enter virtual Ethernet (VE) configuration mode and specify the VLAN.

```
device(config)# interface ve 100
```

12. Assign an IP address and mask and enable the interface.

```
device(config-Ve-100)# ip address 10.2.2.2/24
device(config-Ve-100)# no shutdown
```

13. Confirm the VE configuration.

```
device(config-Ve-100)# do show running-config interface ve 100
interface Ve 100
 ip proxy-arp
 ip address 10.2.2.2/24
 no shutdown
```

Configuring VE route-only mode on a LAG port

Do the following to configure VE route-only mode on a LAG (port-channel) port.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a VLAN.

```
device(config)# vlan 100
```

3. Specify a port-channel interface.

```
device# configure terminal
device(config)# interface port-channel 1
```

4. Enter the **switchport** command to configure Layer 2 characteristics.

```
device(config-Port-channel-1)# switchport
```

5. Specify trunk mode.

```
device(config-Port-channel-1)# switchport mode trunk
```

6. Tag the port to a VLAN.

```
device(config-Port-channel-1)# switchport trunk allowed vlan add 100
```

7. Enable tagging on native VLAN traffic.

```
device(config-Port-channel-1)# switchport trunk tag native-vlan
```

8. Enter the **route-only** command top enable Layer 3 routing exclusively on the port.

```
device(config-Port-channel-1)# route-only
```

   Use the **no route-only** command to revert to default Layer 2 and Layer 3 behavior.

9. Enable the interface and exit to global configuration mode.

```
device(config-Port-channel-1)# no shutdown
device(config-Port-channel-1)# exit
```

10. Verify the port-channel configuration.

```
device(config-Port-channel-1)# do show running-cocnfig interface port-channel 1
 interface Port-channel 1
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 100,200
 switchport trunk tag native-vlan
 route-only
 no shutdown
```

11. Enter virtual Ethernet (VE) configuration mode and specify the VLAN.

```
device(config)# interface ve 100
```

12. Assign an IP address and mask and enable the interface.

```
device(config-Ve-100)# ip address 10.2.2.2/24
device(config-Ve-100)# no shutdown
```

13. Verify the VE configuration.

```
device(config-Ve-100)# )# do show running interface ve 100
interface Ve 100
 ip proxy-arp
 ip address 10.2.2.2/24
 no shutdown
```

Configuring virtual routing interfaces

The Extreme device sends Layer 3 traffic at Layer 2 within a protocol-based VLAN. However, Layer 3 traffic from one protocol-based VLAN to another must be routed. If you want the device to be able to send Layer 3 traffic from one protocol-based VLAN to another on the same device, you must configure a virtual routing interface on each protocol-based VLAN, then configure routing parameters on the virtual routing interfaces.

A *virtual routing interface* is a logical routing interface that the Extreme device uses to route Layer 3 protocol traffic between protocol-based VLANs. It is a logical port on which you can configure Layer 3 routing parameters.

For example, to enable a Extreme device to route IP traffic from one IP protocol VLAN to another, you must configure a virtual routing interface on each IP protocol VLAN, then configure the appropriate IP routing parameters on each of the virtual routing interfaces.

To attach a router interface to a VLAN, using the **router-interface** command:

```
device# configure terminal
device(config)# vlan 2
device(config-vlan-2)# router-interface ve 2
```

NOTE
Only one router VE interface can be mapped to a VLAN. The VLAN ID and the VE ID need not be the same.

Use the **no router interface ve** command to remove the router VE interface.

DHCP Layer 2 relay agent Option 82 for VLANs

This feature supports DHCP Relay Agent Information Option 82 for Layer 2 on VLANs.

NOTE
For a discussion of this feature for Layer 3, refer to the "DHCPv4" chapter in the *Extreme SLX-OS Layer 3 Configuration Guide*.

Overview

It is advantageous to have DHCP relay agent support when DHCP clients and servers are not on the same subnet. DHCP broadcast requests are relayed by the DHCP relay agent to the DHCP server. The DHCP replies are unicasted to the DHCP relay agent, which in turn relays them back to the DHCP client.

For Layer 3, relay agents populate the GIADDR (global IP address) field and also append the "Relay Agent Information" option. DHCP servers use this option for assigning the IP address and other parameters.

In some network configurations that have Layer 2 devices between the DHCP client and the DHCP relay agent, it is better to have Layer 2 relay agents running on Layer 2 devices. This places the agents in a better position to append the Relay Agent Information option because they are closer to the DHCP clients. These agents also broadcast the DHCP messages. (A Layer 3 relay agent, on the other hand, relays the information to the DHCP server.)

Option 82

When this Layer 2 feature is enabled on a VLAN, the Option 82 information is inserted by the Layer 2 relay agent before the DHCP messages are broadcasted further. This information allows the DHCP server to select an IP address or other parameter. The DHCP server echos Option 82 in the reply packets. The DHCP Layer 2 relay agent validates and removes Option 82 and sends the response to the DHCP client.

The format of Option 82 is shown in the following table.

TABLE 6 Option 82 format

Code	Len	Agent Information Field					
82	N	i1	i2	i3	i4	...	iN

NOTE
The length *N* represents the total number of octets in the Agent Information Field. The Agent Information field consists of a sequence of SubOpt/Length/Value tuples for each suboption.

The following table lists the circuit ID suboptions that are added by the relay agent.

TABLE 7 Circuit ID suboptions

Suboption type (1 byte)	Length (1 byte)	VLAN ID <string> (4 bytes)	IF-description string (4 bytes)
2	68		

NOTE
The circuit ID is a combination of the VLAN ID and the interface description string. If the interface description is not configured, the default string "Extreme" is used in the circuit ID.

The following table lists the remote ID suboptions that are added by the relay agent.

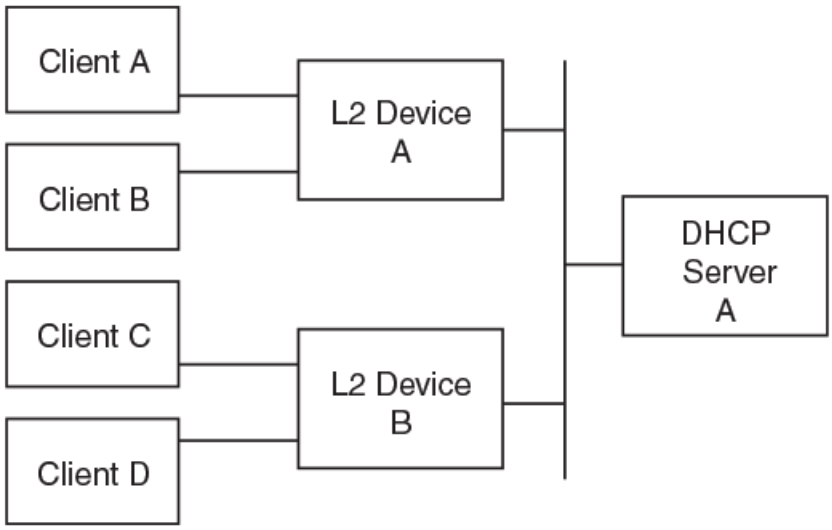
TABLE 8 Remote ID suboptions

Suboption type (1 byte)	Length (1 byte)	VLAN ID (2 bytes)	MAC address (6 bytes)
2	8		

Topologies and operation

The following figure illustrates a single DHCP server and clients on the same subnet.

FIGURE 2 Single DHCP server and clients on same subnet

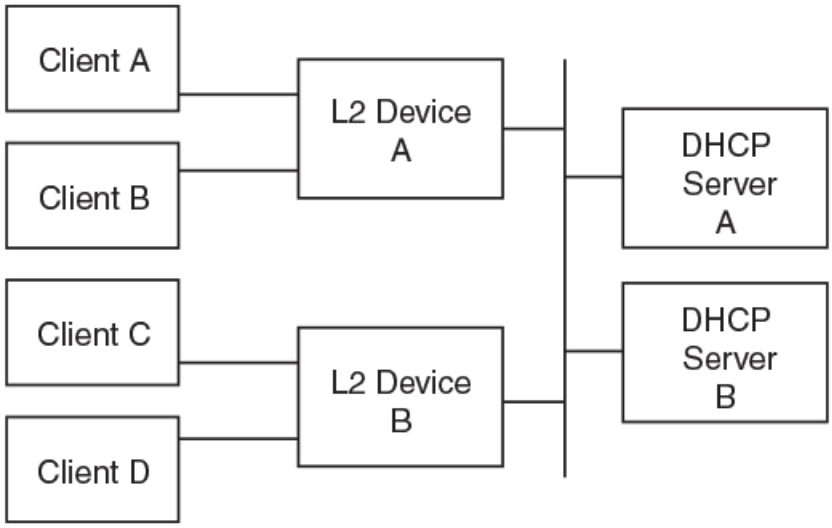


Note the following considerations for the above scenario:

- The Layer 2 relay agent (on the Layer 2 device) appends Option 82 information to a DHCP message received from the client and broadcasts the message to all other ports. The Option 82 information contains the remote ID suboption and the circuit ID suboption.
- The Layer 2 relay agent does not set the GIADDR field.
- The DHCP server echoes the Relay Agent Information option in the response message.
- The Layer 2 relay agent uses the Relay Agent Information option to find out whether it had appended Option 82 to the request message.
- The Layer 2 relay agent removes Option 82 from the packet received from the DHCP server after this option is validated, and then it forwards the message to the interface identified by the Relay Agent Information option.

The following figure illustrates multiple DHCP servers and clients on the same subnet.

FIGURE 3 Multiple DHCP servers and clients on the same subnet

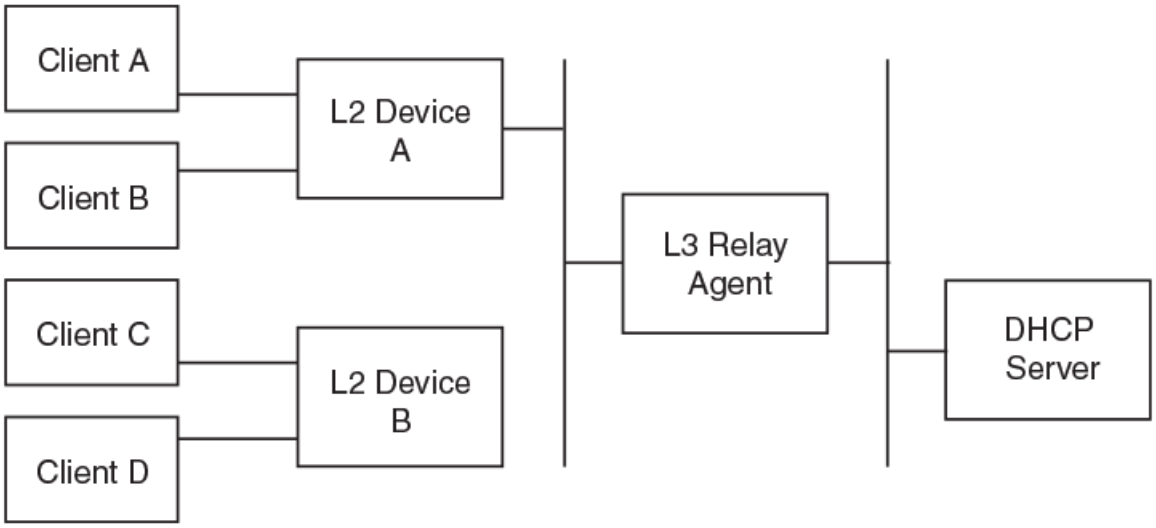


Note the following considerations for the above scenario:

- The Layer 2 relay agent receives multiple responses (from all the servers).
- The processing of DHCP messages in the Layer 2 relay agent remains the same as in the scenario above.

The following figure illustrates a DHCP server on another subnet with clients and one Layer 3 relay agent.

FIGURE 4 DHCP server on another subnet with clients and one Layer 3 relay agent



Note the following considerations for the above scenario:

- The Layer 3 relay agent receives the message relayed by the Layer 2 relay agent. It finds that the message already contains a Relay Agent Information option. The Layer 3 relay agent populates the GIADDR field as appropriate and relays the message to the DHCP server.

- The DHCP server processes the message and unicasts the response to the Layer 3 relay agent on the address specified in the GIADDR field.
- The L3 relay agent processes the response from the server and identifies the outgoing interface. It resets the GIADDR field and broadcasts the message on the identified outgoing interface.

Considerations for Layer 2 relay agent

Note the following:

- All client interfaces are treated as untrusted interfaces.
- There are no separate configurations for the suboptions. Enabling Option 82 for this feature enables the automatic insertion of both the circuit ID and remote ID suboptions.

Configuring DHCP Layer 2 relay agent Option 82

This task enables, verifies, and disables DHCP Relay Agent Information Option 82 for Layer 2 on VLANs.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Specify a VLAN and enter VLAN configuration mode.

```
device(config)# vlan 10
```

3. Enter the **t2 relay agent information option** command.

```
device(config-vlan-10)# l2 relay agent information option
DHCP L2 Relay Agent Information Option is enabled.
```

4. To verify the running configuration on the VLAN, enter the **show running-config** command and specify the VLAN.

```
device# show running-config vlan 10
    vlan 10
    ip dhcp relay information option
```

5. To verify the configuration on the switch, enter the **show ip dhcp relay option** command.

```
device# show ip dhcp relay option
Interface      Circuit-ID               Remote-ID
---------      ----------               ---------
eth0/1         0201630080Extreme        000a0027f8c744e4
eth0/2         0201630081EdgeDevices    00140027f8c744e5
po1000         0201630082Clusters       004b0027f8c744e9
ve10           0201630097Extreme        00640027f8c744f1
```

   You can use the **interface** keyword in the above command to specify an interface.

6. To disable this feature, enter the **no t2 relay agent information option** command.

```
device(config-vlan-10)# no l2 relay agent information option
DHCP L2 Relay Agent Information Option is disabled.
```

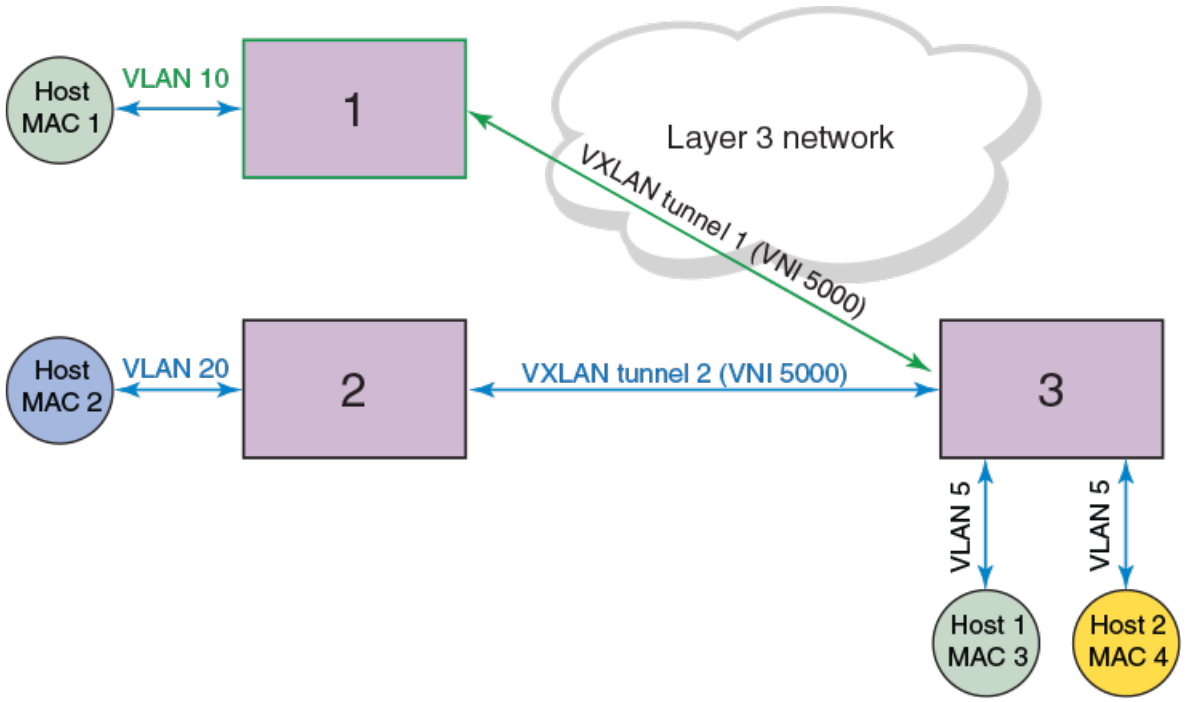
VXLAN Layer 2 Gateway

VXLAN Layer 2 gateway overview

SLX-OS devices can act as a Layer 2 gateway.

The following figure illustrates an example Layer 2 gateway topology.

FIGURE 5 Layer 2 gateway topology



Device	IP address	Mapping
1: SLX 9850 series	1.1.1.1	VNI 5000 < > VLAN 10
2: VDX 6740 series	2.2.2.2	VNI 5000 < > VLAN 20
3: SLX 9540 series	3.3.3.3	VNI 5000 < > VLAN 5

VLANs on each node are extended through common Virtual Network Instance (VNI) 5000. The MAC addresses of local hosts are learned on access points, and MAC addresses are learned on VXLAN tunnels. A split-horizon topology is supported. All nodes participating in the VLAN must be connected through VXLAN tunnels, because there is no tunnel-to-tunnel flooding of broadcast, unknown unicast, and multicast (BUM) traffic.

In this topology, Devices 1, 2, and 3 are VXLAN Layer 2 gateway devices. On Device 3, tunnel 1 and tunnel 2 are mapped to VLAN 5. VLAN 5 has two hosts, MAC 3 and MAC 4. Device 3 is connected to two other hosts, Device 1 and Device 2, which connect to hosts MAC 1 and MAC 2, respectively, through VXLAN tunnels 1 and 2, respectively. If MAC 3 needs to establish traffic to MAC 1, initially there will be BUM flooding and upon a response from MAC 1, MAC 1 is learned through tunnel 1. Subsequent traffic goes directly from MAC 3 to Device 1 on tunnel 1. Traffic in the reverse direction comes from Device 1, is decapsulated, and goes to MAC 3.

VXLAN Layer 2 gateway considerations and limitations

Note the following considerations and limitations for VXLAN Layer 2 gateway.

- A maximum of 512 tunnels are supported.
- A maximum of 8 ECMP paths are supported.
- Layer 2 snooping is not supported.
- VRRPe source IP addresses and EVPN Multi-Chassis Trunks (MCTs) are not supported.
- QoS, TTL, and MTU values are not configurable. The MTU is based on the IP interface MTU. If a packet is bigger than the IP interface MTU minus the VXLAN header, the packet is dropped. The default TTL value is 255. The default QoS value is 0, which is applied to the DSCP field of the IP header.
- VXLAN tunnels have the standard UDP header encapsulated with the standard defined value of 4789. This value is not configurable. SLX-OS expects VXLAN tunnel packets to be received with this value.
- Only the VXLAN extended TCAM profile is supported.
- VXLAN tunnels are not supported when the counter profile 1 or 4 is configured. These profiles do not allocate hardware resources for TX statistics, which is needed for VXLAN tunnels.
- The tunnel TX bytes statistics do not account for the outer VLAN header size.
- When BUM packets are flooded from one tunnel to another, they are expected to be dropped. However, the TX statistics counter on the outbound tunnel increments.

Configuring VXLAN Layer 2 gateway

Follow these steps to configure a VXLAN Layer 2 gateway.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **overlay-gateway** command, specify the name of a gateway, and enter VXLAN overlay gateway configuration mode.

```
device(config)# overlay-gateway GW1
```

3. Enter the **type** command and specify **l2-extension**.

```
device(config-overlay-gw-GW1)# type l2-extension
```

4. Enter the **map vlan vni** command and specify **l2-extension**.

```
device(config-overlay-gw-GW1)# map vlan 5 vni 5000
```

5. Enter the **map bridge-domain** command and specify a bridge domain and VNI.

```
device(config-overlay-gw-GW1)# map bridge-domain 1 vni 2000
```

6. Enter the **ip interface** command and specify a loopback ID.

```
device(config-overlay-gw-GW1)# ip interface loopback 1
```

7. Enter the **site** command, specify a site name, and enter VXLAN overlay gateway site configuration mode.

```
device(config-overlay-gw-GW1)# site mysite1
```

This mode configures a VXLAN tunnel to the site node.

8. Enter the **ip address** command and specify an IP address.

```
device(config-overlay-gw-GW1-site-mysite1)# ip address 1.1.1.1
```

9. Enter the **extend vlan add** command and specify a VLAN.

```
device(config-overlay-gw-GW1-site-mysite1)# extend vlan add 5
```

10. Enter the **extend bridge-domain add** command and specify a bridge domain.

```
device(config-overlay-gw-GW1-site-mysite1)# extend bridge-domain add 1
```

11. Enter the **activate** command to activate the site.

```
device(config-overlay-gw-GW1-site-mysite1)# activate
```

12. In privileged EXEC mode, enter the **show overlay-gateway** command to confirm the gateway configuration.

```
device# show overlay-gateway
Overlay Gateway "GW1", ID 1,
Admin state up
IP address 3.3.3.3 (loopback 1), Vrfdefault-vrf
Number of tunnels 1
Packet count: RX 17909 TX 1247
Byte count : RX (500125) TX 356626
```

13. In privileged EXEC mode, enter the **show tunnel** command to confirm the tunnel configuration.

```
device# show tunnel 61441
Tunnel 61441, mode VXLAN
Ifindex 0x7c00f001, Admin state up, Operstate up
Source IP 3.3.3.3, Vrf: default-vrf
Destination IP 1.1.1.1
Active next hops on node 1:
IP: 4.4.4.5, Vrf: default-vrf
Egress L3 port: Ve45, Outer SMAC: 609c.9f5a.4415
Outer DMAC: 609c.9f5a.0015, ctag: 0
BUM forwarder: yes
```

14. In privileged EXEC mode, enter the **show vlan** command to confirm the VLAN configuration.

```
device# show vlan 5
VLAN             Name            State                      Ports           Classification
(R)-RSPAN                                                   (u)-Untagged
                                                            (t)-Tagged
================ =============== ========================== =============== ====================
5                VLAN05          ACTIVE                     Eth 2/1(t)
                                                            Eth 2/5(t)
                                                            tu61441  vni5000
```

15. In privileged EXEC mode, enter the **show mac-addfress-table** command to confirm the MAC configuration.

```
device# show mac-address-table
VlanId/BDId    Mac-address       Type      State      Ports/LIF/PW
35 (V)         609c.9f5a.5b15    Dynamic   Active     Po 35
45 (V)         609c.9f5a.4415    Dynamic   Active     Po 45
5 (V)          0000.0400.0011    Dynamic   Active     tu61441
5 (V)          0000.0500.0011    Dynamic   Active     Eth 0/5
5 (V)          0000.0400.0011    Dynamic   Active     tu61441
5 (V)          0000.0500.0011    Dynamic   Active     Eth 0/5
Total MAC addresses : 6
device#
```

VXLAN Layer 2 gateway support for bridge domains

An SLX-OS device provides VXLAN Layer 2 gateway support for bridge domains in addition to supporting Layer 2 VLANS. VXLAN gateway support to bridge domains allows a maximum of 4 K bridge domain Virtual Network Interface (VNI) mappings along with a maximum of 4 K VLAN VNI mappings, for a total of 8 K mappings.

Since a bridge domain supports different port and VLAN endpoints, all of its traffic can be extended to a remote node through one VNI.

Also, VXLAN gateway support to bridge domains enables VLAN translation of traffic on both sides of the network. The local VLANs can use different VLAN tags on either side of the network and map to the same VNI.

NOTE
Only point-to-multipoint bridge domains are supported to extend over VXLAN tunnels. Point-to-point bridge domains are not supported.

You can extend the bridge domain under a site configuration. You can configure the bridge domain to VNI mapping automatically with auto mode where the bridge domain to the VNI is mapped implicitly. For example, VLANs 1 through 4096 are mapped to VNI 1 through 4090, respectively, and the bridge domain 1 is mapped to 4097. You can also configure the bridge domain to a VNI map manually, similarly to that of a VLAN.

NOTE
The default tagging mode for a bridge domain is raw mode.

Configuring VXLAN Layer 2 Gateway support for bridge domains

Before performing this configuration, configure a point-to-multipoint bridge domain.

Follow these steps configure a VXLAN Layer 2 gateway to support bridge domains.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a VXLAN overlay gateway and access the overlay gateway configuration mode.

```
device(config)# overlay-gateway gateway1
device(config-overlay-gw-gateway1)#
```

3. Configure the Layer 2 type extension.

```
device(config-overlay-gw-gateway1)# type layer2-extension
```

4. Specify a loopback interface.

```
device(config-overlay-gw-gateway1)# ip interface loopback 1
```

5. Enable the mapping of a bridge domain to a VNI.

```
device(config-overlay-gw-gateway1)# map bridge-domain 1 vni 999
```

6. Create a remote Layer 2 extension site in a VXLAN overlay gateway and access site configuration mode.

```
device(config-overlay-gw-gateway1)# site bd1
```

7. Specify the destination IPv4 address of a tunnel.

```
device(config-site-bd1)# ip address 10.67.67.1
```

8. Configure a bridge domain for the tunnel to the site.

```
device(config-site-bd1)# extend bridge-domain add 1
```

9. Exit site configuration mode.

```
device(config-site-bd1)# exit
```

10. Activate the gateway.

```
device(config-overlay-gateway-gateway1)# activate
```

The following summarizes the configuration example.

```
device# configure terminal
device(config)# overlay-gateway gateway1
device(config-overlay-gw-gateway1)# type layer2-extension
device(config-overlay-gw-gateway1)# ip interface loopback 1
device(config-overlay-gw-gateway1)# map bridge-domain 1 vni 999
device(config-overlay-gw-gateway1)# site bd1
device(config-site-bd1)# ip address 10.67.67.1
device(config-site-bd1)# extend bridge-domain add 1
device(config-site-bd1)# exit
device(config-overlay-gateway-gateway1)# activate
```

VXLAN Layer 2 gateway payload tag processing

An SLX-OS device provides the following modes for the processing of the payload tag that is received on the attachment-circuit packets:

- VXLAN RFC-compliant mode
- Enhanced payload tag transport mode

VXLAN RFC-compliant mode

In RFC-compliant mode, the VLAN tag in a packet is not carried in the packet and must be stripped at the ingress device before the VXLAN encapsulated packet is sent into the network.

To configure RFC-compliant mode, configure the bridge domain in raw mode as shown in the following example.

```
pw-profile test
    vc-mode raw

bridge-domain 10 p2mp
   pw-profile test
```

Then extend the bridge domain in the overlay gateway, as shown in the following example.

```
overlay-gateway gateway1
type layer2-extension
ip interface loopback 1
map bridge-domain 10 vni 999
site vcs1
 ip address 10.67.67.1
 extend bridge-domain add 10
```

NOTE

An SLX-OS device supports RFC-compliant mode with the bridge domain-based VXLAN service only.

Enhanced payload tag transport mode

In enhanced payload tag transport mode, one VLAN tag from the traffic is carried as part of the VXLAN encapsulated packet as an inner payload tag. This tag can carry the PCP value to include the priority information and also can interoperate with other devices. This tag is removed in the remote device capable of this behavior.

NOTE

This mode does not interoperate with RFC-compliant mode.
This mode is supported for VLAN-based VXLAN service and bridge domain-based VXLAN service with tag mode.

To configure enhanced payload tag transport mode, configure the bridge domain in tagging mode as shown in the following example.

```
pw-profile test
    vc-mode tag

bridge-domain 10 p2mp
   pw-profile test
```

Then extend the bridge domain in the overlay gateway, as shown in the following example.

```
overlay-gateway gateway1
type layer2-extension
ip interface Loopback 1
map bridge-domain 10 vni 999
site vcs1
 ip address 10.67.67.1
 extend bridge-domain add 10
```

VXLAN Layer 2 support for LVTEP

This section details support for a logical VXLAN tunnel end point (LVTEP) at Layer 2.

LVTEP control plane

In a BGP EVPN deployment, each node in an LVTEP pair is an independent BGP speaker.

The BGP session between the cluster peers must be configured with an encapsulation of MPLS. The BGP session with the remote peers must be configured with an encapsulation of VXLAN.

The LVTEP control plane uses MCT Control Plane Designated Forwarder Election among the cluster peers. BGP VXLAN tunnels that are discovered automatically are treated as cluster client end points (CCEPs).

The following table shows Ethernet Segment Identifier (ESI) values for the VXLAN tunnels.

TABLE 9 ESI values for VXLAN tunnels

Parameter	Value
ESI type	0
Destination IP address (DIP)	4 bytes
Source IP address (SIP	4 bytes

The ESI label is allocated globally for the LVTEP and is the same for all LVTEP tunnels. The tunnel operational status that is used for LVTEP tunnels is the same as that used for cluster clients.

For BGP EVPN deployment, MAC addresses that are learned on the LVTEP VXLAN tunnels are not synchronized between the cluster peers. However, for static LVTEP, the data-plane MAC addresses that are learned on the CCEPs of the LVTEP tunnel are synchronized between the cluster peers.

> NOTE
> LVTEP is supported only with the TCAM profile VxlanExtended, as configured by the **profile tcam vxlan-ext** command in hardware configuration mode. It is not supported in other TCAM profiles.

All the **show** commands that apply to MCT clients also apply to tunnel cluster clients.

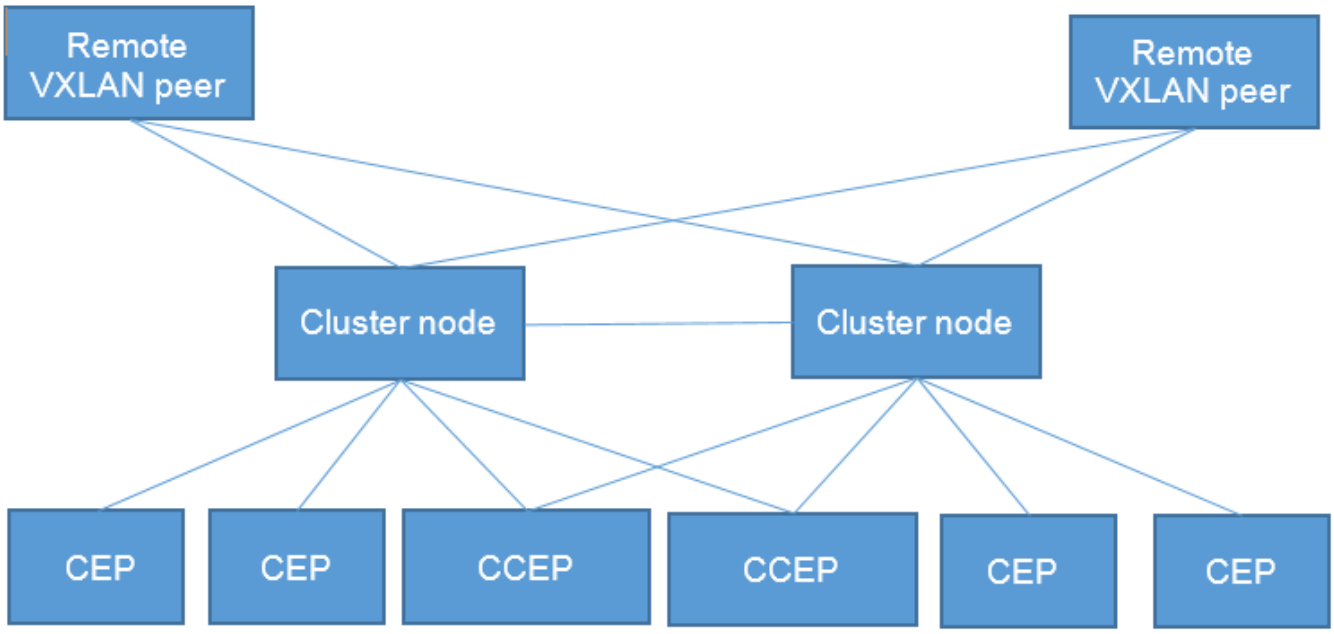
LVTEP data plane

This section presents the details of the LVTEP data plane, including a basic topology, packet formats, and support for various deployment scenarios.

Example topology

The following figure illustrates a basic LVTEP topology for the data plane, with cluster nodes supporting remote peers and client end points (CEPs) and cluster client end points (CCEPs).

FIGURE 6 LVTEP topology



Packet formats

The following tables describe the supporting packet formats.

TABLE 10 VXLAN encapsulation

Outer MAC header	Outer IP header	UDP header	VXLAN header	Original L2 frame

The preceding packet format is used between the remote peers for both unicast and BUM traffic.

TABLE 11 Pseudowire emulation (PWE) Multi-Chassis Trunk (MCT) unicast encapsulation

Outer MAC header	MPLS tunnel label	EVPN unicast label	Original L2 frame

The preceding packet format is used between the cluster peers for unicast traffic.

TABLE 12 PWE MCT multicast encapsulation without ESI label

Outer MAC header	MPLS tunnel label	EVPN multicast label	Original L2 frame

The above packet format is used between the cluster peers for BUM traffic when the traffic is received on the CEP interface.

TABLE 13 PWE MCT multicast encapsulation with ESI label

Outer MAC header	MPLS tunnel label	EVPN multicast label	ESI label	Original L2 frame

The above packet format is used between the cluster peers for BUM traffic when the traffic is received on the CCEP Interface. This applies to both the Layer 2 CCEP interface or the tunnel CCEP interface. The ESI label distinguishes whether the interface is a Layer 2 CCEP interface or a tunnel CCEP interface.

NOTE

For PWE (unicast/multicast) encapsulation, the MPLS tunnel label is present if and only if the peers are not directly connected over an ICL link and there is an MPLS transit router between them. For directly connected ICL peers, there is only the EVPN unicast/multicast encapsulation, with or without an ESI label, and there is no MPLS tunnel label.

Dynamic cluster configuration support

When the cluster is dynamically changed from "Deploy" to "No Deploy" and conversely, the LVTEP tunnel CCEP is accordingly converted to a single VTEP and conversely.

When one or both cluster nodes are configured with "No Deploy" of the cluster configuration, the traffic behavior becomes undeterministic. Recovery scenarios are presented here.

Deploy to No Deploy

Do the following to recover from this error scenario:

1. Deactivate the overlay gateway on both cluster nodes.
2. Undeploy (or remove) the cluster configuration on both the cluster nodes.
3. Change the loopback address on both the cluster nodes to be different from the LVTEP loopback address.
4. Activate the overlay gateway on both the cluster nodes.

No Deploy to Deploy

Do the following to recover from this error scenario:

1. Deactivate the overlay gateway on both the cluster nodes.
2. Change the loopback address on both the cluster nodes to a common LVTEP loopback address that is different from the previous loopback address.
3. Deploy (or configure) a cluster on both the cluster nodes.
4. Activate the overlay gateway on both the cluster nodes.

Deployment scenario 1: BGP session and tunnel down, remote peers

For a BGP EVPN deployment, the MAC addresses learned on the LVTEP tunnel CCEP are flushed. Traffic destined to a remote VXLAN tunnel, although the tunnel may be up on the cluster node, is still flooded on the corresponding VLAN/BD. This behavior differs from that for the Layer 2 CCEP, where traffic is forwarded to the cluster node with the PWE unicast label. The primary reason for this behavior is that the MAC addresses are not synced between the cluster nodes (they are learned on the LVTEP tunnel CCEP).

For the static LVTEP, the behavior is the same as that for the Layer 2 CCEP, where the MAC addresses are synced between the cluster peers. When a BGP session or tunnel down event occurs, the MAC addresses point to the corresponding MCT EVPN PWE and are forwarded to the cluster peer; they are not flooded on the corresponding VLAN/BD.

Deployment scenario 2: BGP session and PW down, cluster peers

The client isolation mode should be configured as loose on both the cluster nodes. This ensures that when the ICL link goes down, the traffic comes from the remote peer.

NOTE
This behavior differs from that for a Layer 2 CCEP and EVPN MPLS, where the nodes operate in loose/strict mode when the ICL link goes down.

Configuring VXLAN LVTEP support

This task configures VXLAN LVTEP support, which includes TCAM profile, loopback, VLAN, Ethernet, and bridge domain (BD), overlay gateway, and MCT cluster configurations.

Configure the appropriate TCAM profile for this feature. Refer to Configuring TCAM profiles to support LVTEP on page 71.

1. Configure multiple loopback interfaces to support BGP neighbor address-family and the LVTEP IP address.
   a) Enter global configuration mode.

```
device# configure terminal
```

   b) In global configuration mode, specify a loopback port number.

```
device(config)# interface loopback 1
```

   c) Configure a loopback interface with OSPF area 0 and an IP address, and enable the interface to support BGP neighbor address-family.

```
device(config-Loopback-1)# ip ospf area 0
device(config-Loopback-1)# ip address 6.6.100.6/32
device(config-Loopback-1)# no shutdown
```

   d) Configure a second loopback interface to support the LVTEP IP address.

```
interface Loopback 2
 ip ospf area 0
 ip address 6.7.100.67/32
 no shutdown
```

   The same address is used for both nodes in the cluster.

2. In global configuration mode, create two VLANs to support a pair of logical interfaces (LIFs) and BDs.

```
device(config)# vlan 11-12
```

3. Configure the LIFs and BDs.
   a) Specify an Ethernet interface.

```
device(config)# interface ethernet 0/5
```

   b) Configure the parent interface as switchport.

```
device(conf-if-eth-0/5)# switchport
```

   c) Specify trunk mode.

```
device(conf-if-eth-0/5)# switchport mode trunk
```

   d) Enable the interface.

```
device(conf-if-eth-0/5)# no shutdown
```

   e) Specify a service instance and enter LIF configuration mode.

```
device(conf-if-eth-0/5)# logical-interface ethernet 0/5.1
```

   f) Specify an interface and create a dual-tagged (inner VLAN) VLAN.

```
device(conf-if-eth-lif-0/5.1)# vlan 10 inner-vlan 1
```

   The VLAN in the LIF configuration is for VLAN tag classification. This example shows a dual-tagged LIF being configured. The expected packet that enters through this port must be dual-tagged, without VLAN 10 and the inner VLAN 1, in order to be classified as a packet received for this LIF.

   g) (Optional) By default, the administrative state of the LIF is "no shutdown." To remove the port from participating in any data traffic without having to shut down the physical interface, enter the **no** form of the **shutdown (LIF)** command.

```
device(conf-if-eth-lif-0/5.1)# no shutdown
```

   h) Repeat Step 3e through Step 3g for the second logical interface, and specify a second inner VLAN.

```
logical-interface ethernet 0/5.2
  vlan 10 inner-vlan 2
```

4. Create and configure a BD.
   a) Create BD 1.

```
device(config)# bridge-domain 1 p2mp
```

   By default, the bridge-domain service type is point-to-multipoint (**p2mp**).
   b) Bind the logical interfaces for attachment circuit (AC) endpoints to the BD.

```
device(config-bridge-domain-1)# logical-interface ethernet 0/5.1
```

   Logical interfaces representing BD endpoints must be created before they can be bound to a BD. For further information, refer to *Logical Interfaces*.
   c) Ensure that local switching is enabled for BD 1.

```
device(config-bridge-domain-1)# local-switching
```

   Local switching is enabled by default.
   d) Enable the dropping of Layer 2 bridge protocol data units (BPDUs) for BD 1.

```
device(config-bridge-domain-1)# bpdu-drop-enable
```

   A default pseudowire (PW) profile is automatically configured, with the following defaults:

```
Vc_mode = RAW Mode
mtu = 1500
mtu_enforce = NO
pw_profile_control_word = 0
pw_profile_flow_label = 0
```

   e) Repeat the above BD configuration for the second BD, as in the following example.

```
bridge-domain 2 p2mp
 logical-interface ethernet 0/5.2
 pw-profile default
 bpdu-drop-enable
 local-switching
```

5. Configure an overlay gateway.
   a) In global configuration mode, specify a gateway.

```
device(config)# overlay-gateway gw1
```

   b) Specify the type as Layer 2 extension.

```
device(config-overlay-gw-gw1)# type layer-2-extension
```

   c) Specify the LVTEP loopback interface.

```
device(config-overlay-gw-gw1)# ip interface loopback 2
```

   d) Configure the automatic mapping of VLANs/BDs to Virtual Network Identifiers (VNIs).

```
device(config-overlay-gw-gw1)# map vni auto
```

   e) Activate the gateway.

```
device(config-overlay-gw-gw1)# activate
```

6. In global configuration mode, enable EVPN configuration mode and configure the EVPN instance.
   a) Enter default EVPN configuration mode.

      ```
      device(config)# evpn
      ```

      Default mode is the only available mode.
   b) Enable the auto-generation of the import and export route-target community attributes for the default EVPN instance.

      ```
      device(config-evpn-default)# route-target both auto
      ```

   c) Enable the auto-generation of a route distinguisher (RD) for the default EVPN instance.

      ```
      device(config-evpn-default)# rd auto
      ```

   d) Add the BDs to the default EVPN instance.

      ```
      device(config-evpn-default)# bridge-domain add 1-2
      ```

   e) Add the VLANs to the default EVPN instance.

      ```
      device(config-evpn-default)# vlan add 11-12
      ```

7. Configure the cluster.
   a) In global configuration mode, specify an MCT cluster name and cluster ID (in this example, "c1" and "1", respectively) to enable cluster configuration mode.

      ```
      device(config)# cluster c1 1
      ```

   b) Ensure that client-isolation loose mode is enabled.

      ```
      device(config-cluster-1)# client-isolation-loose
      ```

      By default, the node with the lower peer IP address is set to the client-isolation mode of loose. Note that loose mode is recommended for LVTEP. This mode ensures that both cluster nodes forward traffic received from the remote peer when the ICL link is down. Without this default configuration, one node becomes loose and one becomes strict (as a result of the peer IP address configuration). The node that becomes strict is not able to forward traffic to the CCEP, because the CCEP links are shut down.
   c) Specify a virtual Ethernet (VE) interface through which to reach the MCT cluster peer.

      ```
      device(config-cluster-1)# peer-interface ve 2
      ```

   d) Specify the IP address of the MCT cluster peer.

      ```
      device(config-cluster-1)# peer 7.7.100.7
      ```

   e) Deploy the cluster.

      ```
      device(config-cluster-1)# deploy
      ```

   f) Exit to Privileged EXEC mode.
8. In Privileged EXEC mode, enter the **cluster management** command and specify a cluster.

   ```
   device# cluster management 2
   ```

   The cluster ID for each node in the cluster must be unique.

9. Configure BGP routing with neighbor and address-family attributes.
   a) In global configuration mode, enable BGP routing and enter BGP router configuration mode.

```
device(config)# router bgp
```

   b) Specify the autonomous system number (ASN) for the AS in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 7.7.100.7 remote-as 100
```

   c) Configure the BGP device to communicate with a neighbor through a specified interface, in this case loopback 1.

```
device(config-bgp-router)# neighbor 7.7.100.7 update-source loopback 1
```

   d) Repeat the above two substeps for the other peer address, as in the following example.

```
 neighbor 8.8.100.8 remote-as 100
 neighbor 8.8.100.8 update-source loopback 1
```

   e) Enable IPv4 and IPv6 unicast address-family.

```
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-router)# address-family ipv6 unicast
```

10. Enable the L2VPN address-family configuration mode to configure a variety of BGP EVPN options.
   a) Enable L2VPN address-family configuration mode and enter BGP EVPN configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

   b) Specify VXLAN encapsulation for the first peer.

```
device(config-bgp-evpn)# neighbor 8.8.100.8 encapsulation vxlan
```

   c) Enable the exchange of information with BGP neighbors and peer groups.

```
device(config-bgp-evpn)# neighbor 8.8.100.8 activate
```

   d) Repeat the above two substeps for the other peer, but specify MPLS encapsulation, as in the following example.

```
neighbor 7.7.100.7 encapsulation mpls
neighbor 7.7.100.7 activate
```

11. Repeat the above steps for the other node in the cluster, with modifications as appropriate.

LVTEP support for other features

This section details how LVTEP supports other features.

AC LIF

AC LIFs of type untagged, single tagged, and double tagged are supported.

Layer 2 ACLs

Layer 2 end points (leaf) support Layer 2 ACLs.

Rate limiting

Layer 2 end points (leaf) do not support rate limiting.

QoS

QoS behavior is similar to that for a single VTEP gateway. Details are listed in the following tables for DiffServe tunneling uniform and pipe modes in an overlay network.

TABLE 14 QoS behavior for DiffServ tunneling uniform mode

Traffic type	VXLAN origination	VXLAN tunnel termination
VLAN cases (untagged)	DSCP 0 and TTL 255 are sent on the tunnel header.	NA
VLAN cases (tagged)	Priority Code Point (PCP) is mapped to IP DCSP of VXLAN tunnel and TTL is set to 255.	DSCP is remapped to PCP of L2 traffic.
BD (raw, single tagged)	PCP of VLAN is mapped to DSCP and TTL is set to 255.	DCSP is remapped to PCP of VLAN.
BD (raw, double tagged)	PCP of outer VLAN is mapped to DSCP and TTL is set to 255.	DCSP is remapped to PCP of both inner and outer VLAN. Original PCP inner VLAN is not retained.
BD (raw, untagged)	DSCP 0 and TTL 255 are sent on the tunnel header	NA
BD (tagged, single tagged)	PCP is mapped to DSCP and TTL is set to 255	DSCP is remapped to PCP of L2 traffic.
BD (tagged, double tagged)	Outer VLAN PCP is mapped to DSCP and TTL is 255 Inner VLAN header is carried with original PCP.	DSCP is remapped to PCP of outer VLAN and Inner VLAN of L2 traffic.
BD (tagged, untagged)	DSCP 0 and TTL 255 are sent on the tunnel header. Dummy VLAN is added: VLAN ID is BD ID and PCP is 0.	NA

TABLE 15 QoS behavior for DiffServ tunneling pipe mode

Traffic type	VXLAN origination	VXLAN tunnel termination
VLAN cases (untagged)	DSCP 0 and TTL 255 are sent on the tunnel header.	NA
VLAN cases (tagged)	DSCP 0 and TTL 255 are sent on the tunnel header.	PCP value is set to 0 as inner VLAN is not carried.
BD (raw, single tagged)	DSCP 0 and TTL 255 are sent on the tunnel header.	PCP value is set to 0 as inner VLAN is not carried.
BD (raw, double tagged)	DSCP 0 and TTL 255 are sent on the tunnel header.	PCP value is set to 0 as inner VLAN is not carried. PCP is cleared for both tags.
BD (raw, untagged)	DSCP 0 and TTL 255 are sent on the tunnel header.	NA
BD (tagged, single tagged)	DSCP 0 and TTL 255 are sent on the tunnel header. VLAN is carried and PCP is carried.	PCP value is set to zero.
BD (tagged, double tagged)	DSCP 0 and TTL 255 are sent on the tunnel header. Inner VLAN is carried and PCP is carried.	PCP value is set to zero.
BD (tagged, untagged)	DSCP 0 and TTL 255 are sent on the tunnel header. Dummy VLAN is added: VLAN ID is BD ID and PCP is 0.	NA

MTU

MTU behavior is similar to that for a single VTEP gateway.

Inner packet tag behavior

Inner packet tag behavior is similar to that for a single VTEP gateway.

The following table summarizes this behavior for VLANs and BDs.

TABLE 16 Inner packet tag behavior for VLANs and BDs

VLAN/BD Configuration	Inner packet tag behavior	Remarks
VLAN	Inner packet is always untagged.	
BD raw mode	Inner packet is always untagged.	This is RFC-compliant behavior.
BD tagged mode	Inner packet is always tagged.	This mode can be used if a tag must always be sent in the inner packet.

Statistics

The LVTEP tunnel CCEP supports statistics. For BUM traffic, although the traffic is suppressed (through split horizon), it is still accounted for as part of the statistics. This behavior is the same as existing single VTEP behavior.

Nondefault TPID

The Tag Protocol Identifier (TPID) is a 16-bit field that identifies the frame as an IEEE 802.1Q-tagged frame. The TPID is set to the default value of 0x8100, but the user can change this value.

The TPID field is located at the same position as the EtherType/length field in untagged frames, and is thus used to distinguish the frame from untagged frames. If you require support for dual tagging or provider backbone bridge (PBB), the outer TPID of the packet must be configured to a value different from the default.

The raw pass-through support for an untagged LIF requires you to configure the interface TPID to a value that allows it to treat all traffic received on that port as untagged. You can configure the TPID on port and LAG interfaces.

> ATTENTION
> When the tag type is changed on interface, the interface is brought down first, causing all learned MAC addresses to be flushed.

High availability (HA) MM failover supports the TPID feature. The TPID configuration is automatically synchronized to the standby module if a standby MM is installed on an SLX-OS device.

Hardware limitations

The TPID feature has the following limitations:

- **Tagging**: The TPID configuration is supported for an outer tag. If dual-tagging needs to be supported on the interface, the inner tag must be 0x8100.
- **TPID**: Up to four TPIDS are supported system-wide. One is used by default (TPID = 0x8100) and cannot be changed.
- **LSP FRR**: Hardware support for LSP FRR is available only for TPID 0x8100. If you require a label switched path with fast reroute (LSP FRR) configuration, note that none of the routable interfaces (whether a router port or a LIF of a VE) can have any nondefault TPID configuration, because FRR always assumes that the link layer has the default TPID of 0x8100.

- **OpenFlow**: If OpenFlow is enabled while the default TCAM profile is used, then no TPID can be configured. This restriction is also enforced in the opposite direction; if any port has TPID configured, then OpenFlow cannot be enabled while the default profile is used.

  NOTE

  The LSP FRR limitation is for any tag-type configured in the device. You can configure either FRR or tag-type on any interface in the device, as in the following example.

```
device(config-router-mpls-lsp-to-avalanche-1)# frr
%Error: Not allowed, when a non-default TPID (tag-type) is configured on any port-channel or
physical interfaces.
device(config-router-mpls-lsp-to-avalanche-1)#
```

Configuring a nondefault TPID

Perform the following steps to configure a nondefault TPID. The interface can be a port or a port-channel (LAG).

1. Do the following to configure a nondefault TPID on an Ethernet interface.
   a) Enter global configuration mode.

```
device# configuration terminal
```

   b) Enter interface configuration mode and specify an Ethernet interface.

```
device(config)# interface ethernet 1/1
```

   c) Use the **tag-type** command to configure the TPID.

```
device(conf-if-eth-1/1)# tag-type 0x9100
```

   By default, all interfaces in the system have default TPID value of 0x8100.

   d) Enter the **show interface ethernet** command to confirm the configuration.

```
device# show interface ethernet 1/1
Ethernet 1/1 is up, line protocol is up (connected)
Hardware is Ethernet, address is 609c.9f5f.5005
    Current address is 609c.9f5f.5005
Pluggable media present
Interface index (ifindex) is 203431936
MTU 1548 bytes
10G Interface
LineSpeed Actual     : 1000 Mbit
LineSpeed Configured : Auto, Duplex: Full
Tag-type: 0x9100
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
Last clearing of show interface counters: 23:12:47
Queueing strategy: fifo
Receive Statistics:
    0 packets, 0 bytes
    Unicasts: 0, Multicasts: 0, Broadcasts: 0
    64-byte pkts: 0, Over 64-byte pkts: 0, Over 127-byte pkts: 0
    Over 255-byte pkts: 0, Over 511-byte pkts: 0, Over 1023-byte pkts: 0
    Over 1518-byte pkts(Jumbo): 0
    Runts: 0, Jabbers: 0, CRC: 0, Overruns: 0
    Errors: 0, Discards: 0
Transmit Statistics:
    0 packets, 0 bytes
    Unicasts: 0, Multicasts: 0, Broadcasts: 0
    Underruns: 0
    Errors: 0, Discards: 0
Rate info:
    Input 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
    Output 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
Route-Only Packets Dropped: 0
Time since last interface status change: 23:12:45
```

   e) To revert to the default TPID value, enter the **no tag-type** command.

```
device(conf-if-eth-1/1)# no tag-type
```

   You can achieve the same result by configuring a tag-type of 0x8100.

2. Do the following to configure a nondefault TPID on a port-channel interface.
   a) Enter global configuration mode.

```
device# configuration terminal
```

   b) Enter interface configuration mode and specify a port-channel interface.

```
device(config)# interface port-channel 20
```

   c) Use the **tag-type** command to configure the TPID.

```
device(config-Port-channel-20)# tag-type 0x88a8
```

   By default, all interfaces in the system have default TPID value of 0x8100.

   d) Enter the **show interface port-channel** command to confirm the configuration.

```
device# show interface port-channel 20
Port-channel 20 is up, line protocol is up
Hardware is AGGREGATE, address is 609c.9f5c.ac07
    Current address is 609c.9f5c.ac07
Interface index (ifindex) is 671088660 (0x28000014)
Minimum number of links to bring Port-channel up is 1
MTU 1548 bytes
LineSpeed Actual     : 300000 Mbit
Allowed Member Speed : 100000 Mbit
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
Tag-type: 0x88a8
Last clearing of show interface counters: 2d01h50m
Queueing strategy: fifo
Receive Statistics:
    34579 packets, 4201368 bytes
    Unicasts: 0, Multicasts: 34579, Broadcasts: 0
    64-byte pkts: 0, Over 64-byte pkts: 17288, Over 127-byte pkts: 17291
    Over 255-byte pkts: 0, Over 511-byte pkts: 0, Over 1023-byte pkts: 0
    Over 1518-byte pkts(Jumbo): 0
    Runts: 0, Jabbers: 0, CRC: 0, Overruns: 0
    Errors: 0, Discards: 0
Transmit Statistics:
    34578 packets, 4201240 bytes
    Unicasts: 0, Multicasts: 34578, Broadcasts: 0
    Underruns: 0
    Errors: 0, Discards: 0
Rate info:
    Input 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
    Output 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
Route-Only Packets Dropped: 0
Time since last interface status change: 1d23h59m
```

Configuring TCAM profiles to support LVTEP

LVTEP configurations require a hardware TCAM profile that supports VXLAN, as illustrated in this task.

ATTENTION
This profile must be applied to all LVTEP configurations in this chapter.

1. Enter global configuration mode and enter the **hardware** command.

```
device# configure terminal
device(config)# hardware
```

2. In hardware configuration mode, enter the **profile tcam** command and specify **vxlan-ext**.

```
device(config-hardware)# profile tcam vxlan-ext
```

3. Save the running configuration to the startup configuration.

```
device# copy running-config startup config
```

4. Reboot the device.

LVTEP show commands

The following **show** commands are helpful in diagnosing a variety of LVTEP configurations.

Cluster show commands

show cluster

```
device# show cluster 1
Cluster c1 1
================
Cluster State: Deployed
Client Isolation Mode: Loose
Configured Member Vlan Range: 11-12
Active Member Vlan Range: 11-12
Configured Member BD Range: 1-2
Active Member BD Range: 1-2
No. of Peers: 1
No. of Clients: 1


 Peer Info:
 ==========
 Peer IP: 7.7.100.7, State: Up
  Peer Interface: Vlan 2


 Client Info:
 ============
 Name            Id           ESI                    Interface                Local/Remote State
 ----          ----       -----------              ---------                ------------------
 tu61441       513        0:0:8:8:64:8:6:7:64:43    Tunnel 61441             Up / Up
```

show cluster client

```
device# show cluster 1 client 513
Client Info:
============
Client: tu61441, client-id: 513, Deployed, State: Up
Interface: Tunnel 61441
Vlans: 11-12
Bridge Domains: 1-2
Number of DF Vlans    : 0
Elected DF for vlans  :
Number of DF Bridge Domains   : 0
Elected DF for Bridge Domains :
```

MCT show commands

show cluster management

```
device# show cluster management

Total Number of Nodes in Cluster   : 2

Node-Id         Switch MAC                  IP Address       Status
-------------------------------------------------------------------
1               60:9C:9F:5A:44:14           4.4.100.4        Connected
2              >60:9C:9F:5A:00:14*          5.5.100.5        Co-ordinator

Please note that 5.5.100.5 (Node-id 2) is this node, 4.4.100.4 (Node-id 1) is remote node
```

BGP show commands

show bgp evpn routes type ethernet-segment detail

```
device# show bgp evpn routes type ethernet-segment detail
Total number of BGP EVPN Ethernet Segment Routes : 2
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
Route Distinguisher: 6.6.100.6:1
1       Prefix: ESR:[00.000808.640806.076443][IPv4:6.6.100.6],  Status: BL,  Age: 1h51m49s
         NEXT_HOP: 0.0.0.0, Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: ExtCom:06:02:06:07:08:08:64:08 ExtCom:06:20:01:04:00:02:00:04
        RT Import 1543:134767624
                esi df_type : 1 expectedNoOfADroutes : 2 flag : 00000004 seqno : 4
            Adj_RIB_out count: 1,  Admin distance 0
            RD: 6.6.100.6:1
Route Distinguisher: 7.7.100.7:1
2       Prefix: ESR:[00.000808.640806.076443][IPv4:7.7.100.7],  Status: BI,  Age: 1h51m48s
         NEXT_HOP: 7.7.100.7, Metric: 1, Learned from Peer: 7.7.100.7 (100)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: ExtCom:06:02:06:07:08:08:64:08 ExtCom:06:20:01:0c:00:02:00:04 ExtCom:03:0c:
00:00:00:00:0a:00
        RT Import 1543:134767624
                esi df_type : 1 expectedNoOfADroutes : 2 flag : 0000000c seqno : 4
            Extended Community: ExtCom: Tunnel Encapsulation (Type MPLS)
            RD: 7.7.100.7:1
```

show bgp evpn routes type autodiscovery detail

```
device# show bgp evpn routes type autodiscovery detail
Total number of BGP EVPN AD Routes : 4
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
Route Distinguisher: 6.6.100.6:1
1       Prefix: AD:[00.0008086408060076443][4294967295],  Status: BL,  Age: 1h52m34s
         NEXT_HOP: 0.0.0.0, Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: ExtCom:06:01:00:00:01:32:0c:00 RT 100:1073741835 RT 100:1073741836
            ESI Label Ext Community: 20057088     All-Active (0x00000000)
            Adj_RIB_out count: 1,  Admin distance 0
            RD: 6.6.100.6:1
Route Distinguisher: 6.6.100.6:33
2       Prefix: AD:[00.0008086408060076443][4294967295],  Status: BL,  Age: 1h52m34s
         NEXT_HOP: 0.0.0.0, Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: ExtCom:06:01:00:00:01:32:0c:00 RT 100:1073745921 RT 100:1073745922
            ESI Label Ext Community: 20057088     All-Active (0x00000000)
            Adj_RIB_out count: 1,  Admin distance 0
            RD: 6.6.100.6:33
Route Distinguisher: 7.7.100.7:1
3       Prefix: AD:[00.0008086408060076443][4294967295],  Status: BI,  Age: 1h52m36s
         NEXT_HOP: 7.7.100.7, Metric: 1, Learned from Peer: 7.7.100.7 (100)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: ExtCom:06:01:00:00:01:32:0c:00 RT 100:1073741835 RT 100:1073741836 ExtCom:
03:0c:00:00:00:00:0a:00
            ESI Label Ext Community: 20057088     All-Active (0x00000000)
            Extended Community: ExtCom: Tunnel Encapsulation (Type MPLS)
            RD: 7.7.100.7:1
Route Distinguisher: 7.7.100.7:33
4       Prefix: AD:[00.0008086408060076443][4294967295],  Status: BI,  Age: 1h52m36s
         NEXT_HOP: 7.7.100.7, Metric: 1, Learned from Peer: 7.7.100.7 (100)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: ExtCom:06:01:00:00:01:32:0c:00 RT 100:1073745921 RT 100:1073745922 ExtCom:
03:0c:00:00:00:00:0a:00
            ESI Label Ext Community: 20057088     All-Active (0x00000000)
            Extended Community: ExtCom: Tunnel Encapsulation (Type MPLS)
            RD: 7.7.100.7:33
```

show bgp evpn routes type inclusive-multicast detail

```
device# show bgp evpn routes type inclusive-multicast detail
Total number of BGP EVPN IMR Routes : 12
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
Route Distinguisher: 6.6.100.6:32779
1       Prefix: IMR:[0][IPv4:6.6.100.6],  Status: BL,  Age: 1h54m39s
         NEXT_HOP: 0.0.0.0, Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: RT 100:1073741835
            Adj_RIB_out count: 2,  Admin distance 0
            L2 Label: 11
            RD: 6.6.100.6:32779

Route Distinguisher: 7.7.100.7:32779
5       Prefix: IMR:[0][IPv4:7.7.100.7],  Status: BI,  Age: 1h53m37s
         NEXT_HOP: 7.7.100.7, Metric: 1, Learned from Peer: 7.7.100.7 (100)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: RT 100:1073741835 RT 100:268435456 RT 100:0 ExtCom:03:0c:00:00:00:00:0a:00
            PMSI Attribute Flags: 0x00000000  Label-Stack: 817163 Tunnel-Type: 6 Tunnel-IP: 0.0.0.0
            Extended Community: ExtCom: Tunnel Encapsulation (Type MPLS)
            L2 Label: 817163
            RD: 7.7.100.7:32779

Route Distinguisher: 8.8.100.8:32779
9       Prefix: IMR:[0][IPv4:8.8.100.8],  Status: BI,  Age: 1h53m29s
         NEXT_HOP: 8.8.100.8, Learned from Peer: 8.8.100.8 (100)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: RT 100:1073741835 RT 100:268435467 RT 100:11 ExtCom:03:0c:00:00:00:00:08:00
            PMSI Attribute Flags: 0x00000000  Label-Stack: 11 Tunnel-Type: 6 Tunnel-IP: 8.8.100.8
            Extended Community: ExtCom: Tunnel Encapsulation (Type Vxlan)
            L2 Label: 11
            RD: 8.8.100.8:32779
```

show bgp evpn routes type mac detail

```
device# show bgp evpn routes type mac detail
Total number of BGP EVPN MAC Routes : 2
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
Route Distinguisher: 6.6.100.6:36865
1       Prefix: MAC:[0][0000.0607.0000],  Status: BL,  Age: 0h0m53s
         NEXT_HOP: 0.0.0.0, Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: RT 100:1073745921
            Adj_RIB_out count: 2,  Admin distance 0
            L2 Label: 4097
            ESI : 00.000000000000000000
            RD: 6.6.100.6:36865
Route Distinguisher: 8.8.100.8:32779
2       Prefix: MAC:[0][0000.0807.0000],  Status: BI,  Age: 0h5m16s
         NEXT_HOP: 8.8.100.8, Learned from Peer: 8.8.100.8 (100)
          LOCAL_PREF: 100,  MED: 0,  ORIGIN: incomplete,  Weight: 0
         AS_PATH:
            Extended Community: RT 100:1073741835 RT 100:268435467 RT 100:11 ExtCom:03:0c:00:00:00:00:08:00
            Extended Community: ExtCom: Tunnel Encapsulation (Type Vxlan)
            L2 Label: 11
            ESI : 00.000000000000000000
            RD: 8.8.100.8:32779
```

Tunnel show commands

show tunnel brief

```
device# show tunnel brief
Tunnel 61441, mode VXLAN, node-ids 1
Admin state up, Oper state up
Source IP 6.7.100.67 (  Loopback 2 ), Vrf default-vrf
Destination IP 8.8.100.8
```

show tunnel

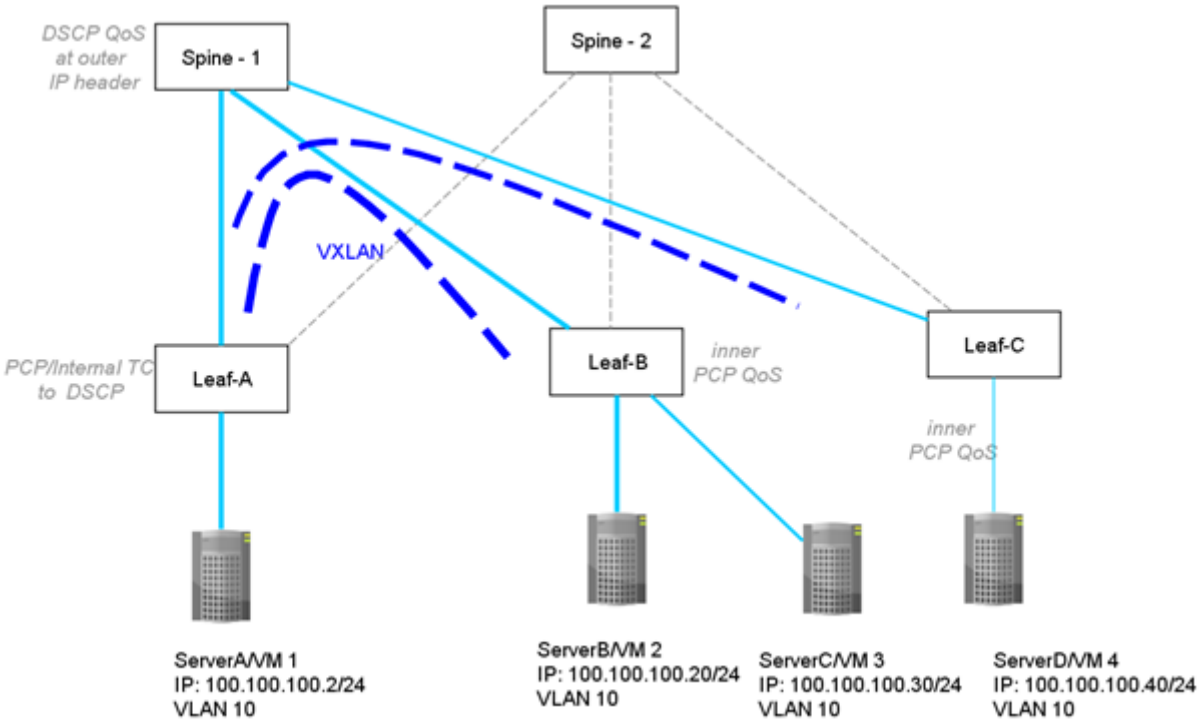
```
device# show tunnel 61441
Tunnel 61441, mode VXLAN, node-ids 1
Ifindex 0x7c00f001, Admin state up, Oper state up
Overlay gateway "gw1", ID 1
Source IP 6.7.100.67 (  Loopback 2 ), Vrf default-vrf
Destination IP 8.8.100.8
Configuration source BGP-EVPN
MAC learning BGP-EVPN
Tunnel QOS mode UNIFORM
Active next hops on node 1:
    IP: 10.6.8.8, Vrf: default-vrf
    Egress L3 port: Ve 3, Outer SMAC: 609c.9f5a.3d15
    Outer DMAC: 609c.9f5a.4515, ctag: 0
    BUM forwarder: yes

Packet count: RX 167993610       TX 995395
Byte count  : RX 29902862580     TX 226950060
```

QoS for VXLAN Layer 2 gateways

SLX-OS VXLAN Layer 2 gateways can support QoS.

A VXLAN L2 gateway can interconnect tenants in the same subnet through the VXLAN configuration. The following figure illustrates a simple use case where a packet is sent from VM1. If it is a BUM packet, Leaf-A floods through the packet to all the VTEPs in the same VXLAN segment, until the MAC table at Leaf-A is populated with corresponding entries through mechanisms such as EVPN. Meanwhile, the known unicast packet is forwarded in unicast mode to the corresponding VTEP only. At Leaf-A, the Traffic Class/802.1p marking of the tenant frame determines the DSCP of the added IP header, and the IP header is followed by the VXLAN header. At terminating leaf nodes, the DSCP of the IP header is ignored and the Traffic Class/802.1p marking of the tenant frame is thereby exposed, and the original tenant frame is used to determine the QoS policy for switching the frame to the destination VM.

FIGURE 7 VXLAN L2 gateway interconnection of tenants in the same subnet



Configuring QoS for VXLAN Layer 2 gateways

Enter the **qos-ttl-mode uniform** command to set the QoS type mode to uniform, which is the default, when configuring the VXLAN L2 gateway. For example, enter the following commands when configuring the gateway for the ingress (tunneling) side of packet forwarding:

```
device# configure terminal
device(config)# overlay-gateway gateway_L2
device(config-overlay-gw-gateway_L2)# type layer2-extension
device(config-overlay-gw-gateway_L2)# ip interface loopback 1
device(config-overlay-gw-gateway_L2)# qos-ttl-mode uniform
```

```
device(config-overlay-gw-gateway_L2)# map vni auto
device(config-overlay-gw-gateway_L2)# activate
```

NOTE

For information about QoS configuration on VXLAN L3 gateways and on VLAN L2 and L3 gateway interconnections, refer to "QoS for VXLAN Layer 3 gateways" and "QoS for VXLAN Layer 2 and Layer 3 gateway interconnections".

VXLAN Layer 3 Gateway

Overview

SLX routers support VXLAN Layer 2 gateway functionality. By acting as VXLAN Layer 3 gateways, SLX routers are capable of routing Layer 3 traffic while also terminating VXLAN tunnels.

To support Layer 3 functionality, a virtual Ethernet (VE) interface must be configured over either a VLAN or a bridge domain (BD) that contains both VXLAN tunnel members and attachment circuit (AC) end-point members. Such a VE (also known as "VE over VXLAN" or "VXLAN VE") can route and switch VXLAN traffic simultaneously.

With VXLAN Layer 3 gateways over VLANs/BDs, both static/EVPN and single/logical, the following options are supported:

- Single VTEP, static VLAN
- Single VTEP, static BD
- Single VTEP, EVPN VLAN
- Single VTEP, EVPN BD
- Logical VTEP, EVPN VLAN (default VRF only)
- Logical VTEP, EVPN BD (default VRF only)

The following table lists and describes the support for a variety of functionalities available under VXLAN Layer 3 gateway.

TABLE 17 VXLAN Layer 3 gateway support for functionalities

Functionality	Description	Comments
Routing protocols	Routing protocols cannot be enabled on a VE configured as a VXLAN Layer 3 gateway.	No routing protocols (such as OSPFor ISIS) are supported on such a VE.
VRF: VRF-lite/Multi-VRF)	A VE over VXLAN can be part of a nondefault VRF.	L3VPN-VRF is not yet supported under Logical VTEP.
ECMP	Support ECMP paths (8) for tunnel routing.	More than eight is not restricted but is not supported. The topology must be such that more than eight paths are not present for a VXLAN tunnel.
Statistics	Tunnel statistics are supported.	By default, statistics are enabled for both directions. If hardware resources are not available, then "N/A" is displayed.
BFD	BFD is not supported.	BFD is not supported for static tunnels.
VRRP	VRRPe source IP address/EVPN-MCT is not supported.	CLI configuration is not restricted.
MTU	MTU value is not configurable.	MTU is based on an IP interface MTU. If the packet is bigger than the IP interface MTU minus the VXLAN header, the packet is dropped.
TTL	TTL value is not configurable.	Default TTL value is 64.

TABLE 17 VXLAN Layer 3 gateway support for functionalities (continued)

Functionality	Description	Comments
QoS	QoS is not configurable.	Default QoS value is 0, which is applied to the DSCP field of the IP header.
-Tunnel uniform/pipe mode for TTL/QoS	Tunnel mode (uniform or pipe mode) is not configurable	By default, tunnel mode is pipe mode: in both directions, DSCP/TTL values are not carried from or to the native packet. In the VLAN-to-VXLAN direction, the default tunnel QoS/TTL values are used.
Exporting VE-over-VXLAN interface IP address using other protocols	Routing protocols running on other IP interfaces can export the VE-over-VXLAN IP address as connected routes.	A VE with VXLAN tunnels is treated as a directly connected subnet. This VE does not support protocols, as described above. However, as there is a connected subnet, reachability to this VE can be advertised through protocols such as OSPF, ISIS, and so on, configured as part of other Layer 3 interface configurations.
Ping/Traceroute	Ping and Traceroute are supported.	Ping supports traffic from/to VXLAN tunnels.
ARP	Dynamic ARP learning is supported in the VXLAN VE.	
Proxy ARP	Proxy ARP is not supported.	Proxy ARP configuration is not restricted, but the functionality is not supported.
Static ARP	Static ARP is supported.	Static ARP to an IP address reachable through a VXLAN tunnel is supported. The interface in the static ARP must be configured as the VE interface to which the host on the VXLAN tunnel is connected.
IPv6	IPv6 is supported.	
Static routes	Static routes are supported.	A static route can be configured to an IP address that is reachable through a VXLAN tunnel.
RPF	Reverse path forwarding (RPF) is not supported in the VXLAN VE.	RPF configuration is not restricted, but RPF functionality is not supported.
Multicast	Layer 3 multicast is not supported.	
PBR	Policy-based routing (PBR) is not supported.	ACL/PBR for native packets is not supported.
HA/ISSU	Hitless HA/ISSU is not supported.	Traffic hits are observed.
Inter-overlay routing	• IP routing is allowed from VE over VXLAN to VLAN-VE and vice versa. • IP routing is allowed from one VE-over-VXLAN tunnel to another. • IP routing is allowed between any other tunnel type (such as GRE/IP tunnel/MPLS-based pseudowire tunnels) to a VE-over-VXLAN tunnel. • Inter-VRF routing is not supported.	
Interoperability	The Layer 3 gateway interoperates with other SLX platforms in extension mode.	Interoperability with other VXLAN-supporting devices/hypervisors is not restricted but is not supported.
CAM profile	VXLAN L2/L3 gateway is supported only in the VxlanExtended TCAM profile	The configuration is not restricted in other profiles, but functionality is not supported.
Layer 2 gateway functionality	All the present Layer 2 gateway functionalities are supported on the Layer 3 gateway.	Only static VXLAN tunnels with regular VTEPs are supported; logical VTEP tunnels are not supported.

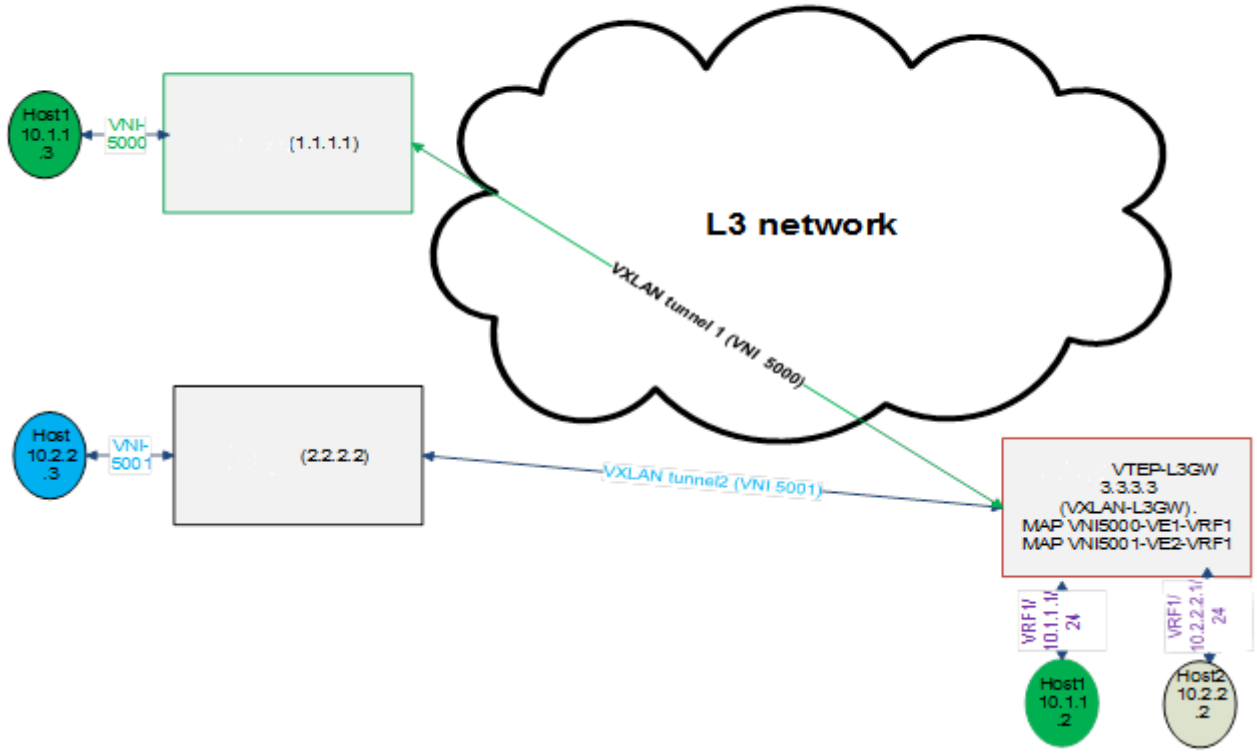
VXLAN Layer 3 gateway modes

Single-VTEP static VLAN/VXLAN-VE Layer 3 gateway

This is the most basic mode of VXLAN Layer 3 gateway (L3GW). In this mode of the L3GW, the VE is configured directly on the VLAN associated with the VXLAN Virtual Tunnel End Point (VTEP), which is mapped to the tunnel VXLAN Network Identifier (VNI).

One or more attachment circuit (AC) endpoints can be associated with the VLAN that is mapped to the VXLAN VE, or the VLAN can just contain the tunnel as its only member. The VXLAN tunnel is static, meaning that it is configured manually. The following figure illustrates a static L3GW topology.

FIGURE 8 Static L3GW topology



The scenario for this topology is as follows:

- A customer has two subnets, 10.1.1.0/24 and 10.2.2.0/24, which have hosts 10.1.1.3 (VNI 5000) and 10.2.2.3 (VNI 5001), respectively. The VRF of the customer is configured as VRF1 on the SLX nodes.
- Also on the SLX nodes, VLAN 10 (configured with routing VE1) maps to VNI 5000, and VLAN 20 (configured with routing VE2) maps to VNI 5001.
- Host 10.1.1.2 (Ethernet 1/1, VLAN10) maps to VNI 5000, and Host 10.2.2.2 (Ethernet 1/2, VLAN 20, maps to VNI 5001).

- If Host 10.1.1.3 must communicate with Host 10.2.2.2, packets must be routed at the VRF1 level. Packets come in on VXLAN tunnel 1 (VNI 5000) and are decapsulated and sent to interface Ethernet 1/1 upon ARP resolution, or an ARP resolution is attempted if the ARP is not resolved.

Single-VTEP static BD/VXLAN-VE

This scenario is the same as for single-VTEP static VLAN/VXLAN-VE L3GW, but in this case the VE is configured on a bridge domain (BD) that is extended over the VXLAN tunnel.

EVPN-based VXLAN Layer 3 gateway

EVPN VXLAN-L3GW functionality support is one of the several key features under the larger umbrella of IP Fabrics.

The following sections describe two features for Layer 3 routing.

IP-MAC routes on a single VTEP

Similar to the normal MAC routes that are exported and installed by EVPN BGP extensions, IP-MAC routes are also exported and installed on the remote nodes. The components of this scenario are detailed here.

BGP MAC/IP routes

This kind of route represents L3-to-L2 mapping, which is basically through ARP or ND. Static, dynamic ARP/ND entries are both exported to remote PEs and get installed as host routes. IPv4/IPv6 addresses that are configured on VE interfaces are also exported.

Upon ARP learning/gleaning/snooping on a local PE, ARP/ND information is exported to its EVPN BGP peers. The information mainly includes the following: MAC, IP/IPv6, L2-VNI, and L3-VNI. Such imported ARP/ND routes are installed or withdrawn as host routes in the hardware on the remote nodes. In the control plane they are available through the ARP suppression cache, which could be further used to reply for further ARP requests from hosts attached to the remote PE.

The packet path is as follows:

- When traffic that is bound to a remote host is received on the ingress PE GW, no ARP request is generated, as the route table already has the hardware host entry to forward or route the traffic to the destined host. This situation prevents ARP flooding and further processing.
- Packets get routed on the ingress PE itself, and then are switched all the way to the destination host, through the egress PE. Because routing occurs on the ingress PE and switching occurs on the remote PE, this type of forwarding is also termed "asymmetric routing."

On the nondefault VRF, the ARP/ND exports can have two subscenarios, depending on whether L2-VNI is extended on that PE or not:

- The imported IP-MAC route can be resolved against the L2-VNI if the L2-VNI is extended over the VXLAN tunnel, in which case the packet path is similar to that described previously.
- If the L2-VNI is not extended over the tunnel on that PE, the IP-MAC route is resolved against the L3-VNI. In this case the packet path followed is similar to a prefix routes path using L3-VNI, as described in the following section.

On the default VRF, formal host IP forwarding is done.

Layer 3 VNI on a single VTEP

For multitenant scenarios using VRFs in data centers, the L3-VNI identifies a particular tenant VRF across a VXLAN-EVPN tunnel. As the name suggests, L3-VNI is used mainly for routing purposes and in short identifies the tenant VRF.

BGP IP prefix routes on VRFs are exported to the remote PE by means of EVPN (Type-5). The information mainly includes the following: Egress-PE-GW-MAC, IP/IPv6 prefix route, and L3-VNI.

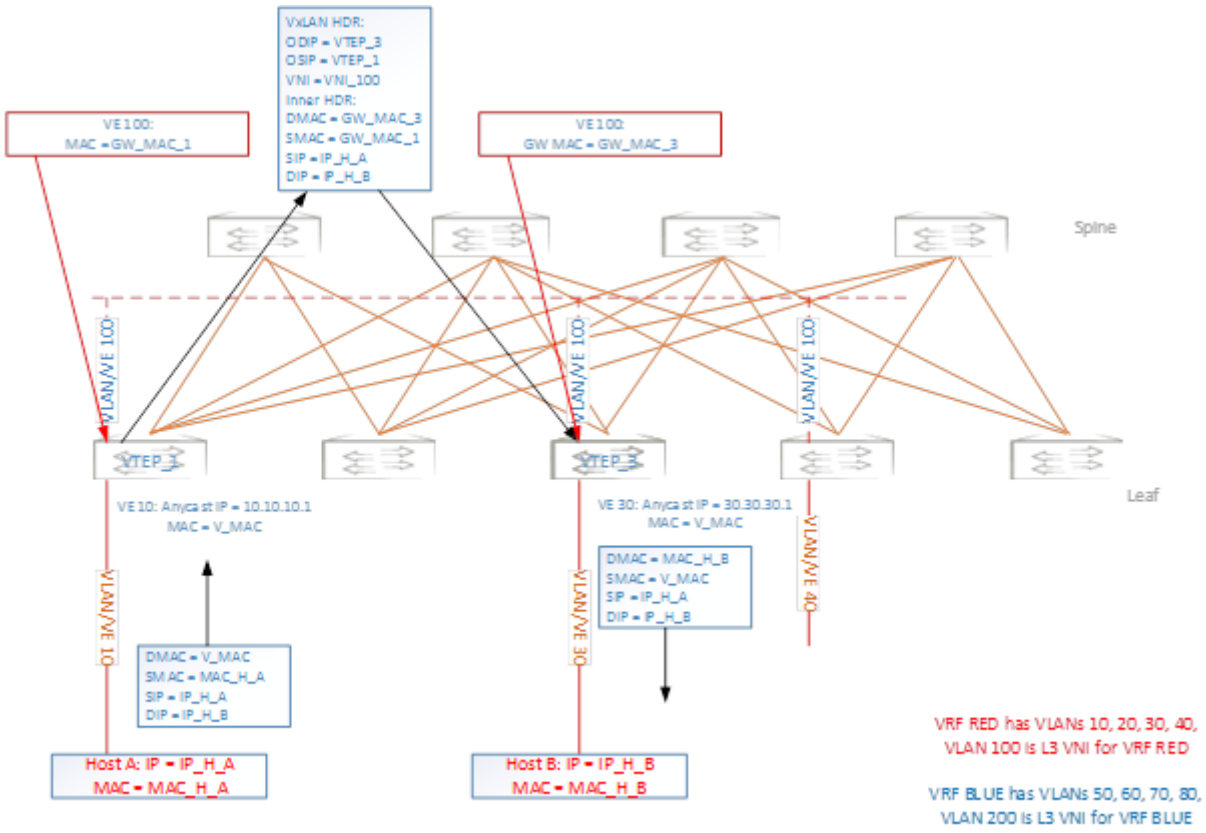
Such imported IP prefix routes are imported to VRFs and are installed as VRF routes, with the VXLAN tunnel having L3-VNI as the outgoing port and remote PE-GW-MAC as the destination MAC within the inner payload L2 header.

The packet path is as follows:

- The Layer 3 routing traffic that is originated on a particular VRF is terminated on the ingress PE gateway. As part of the L3 routing within the tenant VRF on the Ingress PE, the L3 packet is carried over the VXLAN tunnel to the egress PE by means of L3-VNI. The payload packet (L3) is always marked with the egress PE as the next hop.
- When the packet arrives at the egress PE, the outer header L3-VNI is used as an identifier to the tenant VRF, and the inner packet gets routed within this tenant VRF context.
- Because routing takes place on both the ingress and egress PE, this routing is also termed "symmetric routing."

The following figure illustrates this topology.

FIGURE 9 Layer 3 VNI on a single VTEP

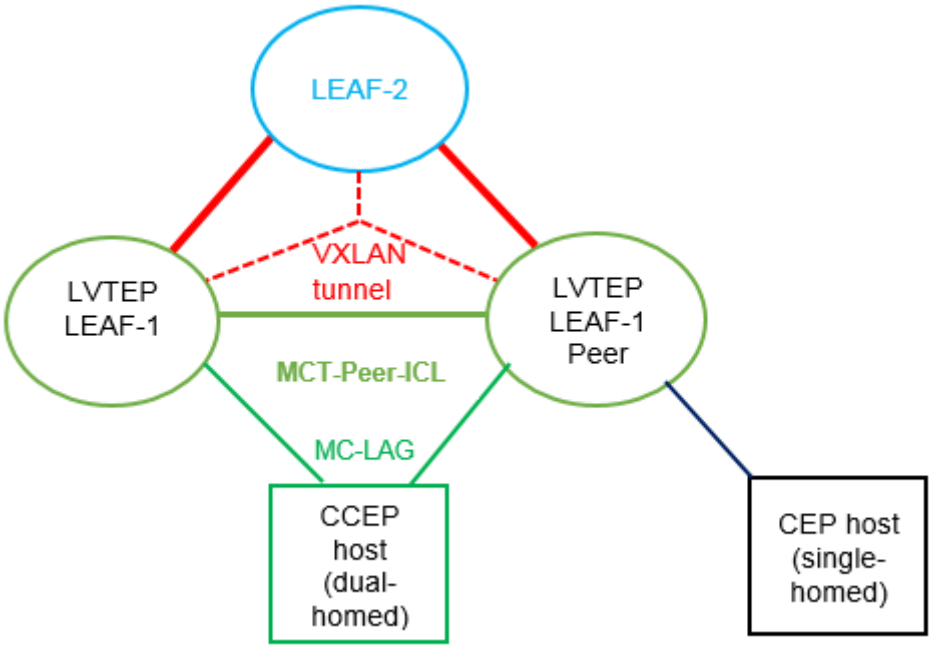


EVPN-based VXLAN Layer 3 gateway on LVTEP

This section discusses the Layer 3 functionality support on such a logical VTEP (LVTEP).

The following figure illustrates a VXLAN LVTEP topology.

FIGURE 10 VXLAN LVTEP topology



The LVTEP is formed through MCT peering (spoke-PW-peer) between Leaf-1 and its peer node, Leaf-12-peer, to provide redundancy for a VXLAN leaf node.

A VXLAN tunnel is created between such an LVTEP leaf and a remote leaf. The source IP address of the VXLAN tunnel is the same on both nodes. Therefore, the tunnel has a single tunnel representation on the remote leaf (Leaf-2 in the figure). The logical connection of the tunnel is shown as the dotted red line.

The single tunnel on Leaf-2 has two underlay paths to reach Leaf-1 and the Leaf-1 peer. Any traffic southbound from Leaf-2 is load balanced and can end up in either of the LVTEP peers.

IP-MAC routes on LVTEP

Similar to the single-VTEP case, the normal MAC-IP routes are exported and installed by EVPN BGP extensions on LVTEP between the leaf nodes, providing for the following behavior:

- Only one of the LVTEP peers that learns ARP (source LVTEP node) exports the route to the remote leaf. Such an imported route is installed as a host route that is pointed to the VXLAN tunnel, which has two underlay paths.
- The source LVTEP also syncs the ARP route to its LVTEP peer over ICL as part of MCT. These routes are installed pointing to the ICL interface (PW). Such synced IP-MAC routes are not readvertised to the VXLAN peer.
- The remote leaf exports its IP-MAC routes to both the LVTEP peers, and both peers install the routes in hardware—as host routes pointing to the local VXLAN tunnel toward the remote leaf.

A BGP MAC/IP route represents L3-to-L2 mapping, which is basically ARP or ND. Static and dynamic ARP/ND entries are exported to remote PEs and get installed as host routes. IPv4/IPv6 addresses that are configured on VE interfaces are also exported.

Upon ARP learning/gleaning/snooping on a local PE, ARP/ND information is exported to its EVPN BGP peers. The information mainly includes the following: MAC, IP/IPv6, L2-VNI, L3-VNI, and ESI segment. (In VXLAN, the ESI segment ID is always 0.)

Such imported ARP/ND routes are installed or withdrawn as host routes in the hardware on the remote nodes. In the control plane they are available through the ARP suppression cache, which can be further used to reply for further ARP requests from hosts that are attached to the remote PE.

The packet path is as follows:

- When traffic bound to a remote host is received on the Ingress PE GW, no ARP request is generated, as the route table already has the hardware host entry to forward/route the traffic to the destined host. This situation prevents ARP flooding and further processing.
- Packets get routed on the ingress PE itself, and then are switched all the way to the destination host, through the egress PE. Because routing occurs on the ingress PE and switching on the remote PE, this type of forwarding is also termed "asymmetric routing."

On the nondefault VRF, the ARP/ND exports can have two subscenarios, depending on whether L2-VNI is extended on that PE or not:

- The imported IP-MAC route could be resolved against L2-VNI if L2-VNI is extended over the VXLAN tunnel, in which case the packet path is similar to the one described previously.
- If L2-VNI is not extended over tunnel on that PE, the IP-MAC route is resolved against L3-VNI, in which case the packet path followed is similar to the prefix routes path using L3-VNI.

On the default VRF, normal host IP forwarding always occurs.

Configuring VXLAN Layer 3 gateway

Configuring TCAM profiles to support Layer 3 gateway

Layer 3 gateway requires a hardware TCAM profile that supports VXLAN, as illustrated in this task.

ATTENTION
This profile must be configured for all the remaining tasks in this chapter.

1. Enter global configuration mode and enter the **hardware** command.

```
device# configure terminal
device(config)# hardware
```

2. In hardware configuration mode, enter the **profile tcam** command and specify **vxlan-ext**.

```
device(config-hardware)# profile tcam vxlan-ext
```

3. Save the running configuration to the startup configuration.

```
device# copy running-config startup config
```

4. Reboot the device.

Configuring a single-VTEP static VLAN/VE Layer 3 gateway

Follow these steps to configure a single-VTEP static VLAN/VE Layer 3 gateway.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 85.

2. Configure a loopback address.

```
interface Loopback 100
 no shutdown
 ip address 40.40.40.1/32
```

3. Configure an IP interface to be the interface of the VXLAN tunnel.

```
interface Ethernet 0/10
 ip proxy-arp
 ip address 50.50.50.1/24
 no shutdown
```

4. Create a VLAN.

```
vlan 500
```

5. Configure an attachment circuit (AC) endpoint.

```
interface Ethernet 0/20
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 500
 switchport trunk tag native-vlan
 no shutdown
```

6. Configure the VTEP.

```
overlay-gateway test
 type layer2-extension
 ip interface Loopback 100
 map vlan 500 vni 15000
 activate
 site VCS_2
 ip address 40.40.40.2  << This must be the remote-end loopback IP address and be reachable
 extend vlan add 500
```

7. Configure VE over VXLAN by configuring a VE over the VLAN associated with VXLAN.

```
vlan 500
 router-interface ve 500

int ve 500
 ip address 15.15.15.1/24
 ipv6 address 1001::1/64
 no shutdown
```

Configuring a single-VTEP static BD/VE Layer 3 gateway

Follow these steps to configure a single-VTEP static BD/VE Layer 3 gateway.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 85.
2. Configure a loopback address.

```
interface Loopback 100
 no shutdown
 ip address 40.40.40.1/32
```

3. Configure an IP interface to be the interface of the VXLAN tunnel.

```
interface Ethernet 0/10
 ip proxy-arp
 ip address 50.50.50.1/24
 no shutdown
```

4. Create a VLAN.

```
vlan 500
```

5. Configure an attachment circuit (AC) endpoint with a logical interface.

```
interface Ethernet 0/20
 switchport
 switchport mode trunk-no-default
 logical-interface eth 0/20.500 vlan 500
 no shut
```

6. Configure a bridge domain (BD).

```
bridge-domain 500 p2mp
 logical-interface eth 0/20.500
 pw-profile default
 bpdu-drop-enable
 local-switching
```

7. Configure the VTEP.

```
overlay-gateway test
 type layer2-extension
 ip interface Loopback 100
 map vlan 500 vni 15000
 activate
 site VCS_2
 ip address 40.40.40.2  << This must be the remote-end loopback IP address and be reachable
 extend bridge-domain add 500
```

8. Configure VE over VXLAN by configuring a VE over the VLAN that is associated with VXLAN.

```
vlan 500
 router-interface ve 500

int ve 500
 ip address 15.15.15.1/24
 ipv6 address 1001::1/64
 no shutdown
```

Configuring an EVPN Layer 3 gateway for MAC IP routes

Do the following to configure a Layer 3 gateway for MAC IP routes.

This configuration is similar to that for Layer 2 EVPN MAC routes.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 85.
2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
  local-as 100
    neighbor 98.0.0.1 remote-as 100
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  address-family l2vpn evpn
    graceful-restart
    neighbor 98.0.0.1 encapsulation vxlan
    neighbor 98.0.0.1 activate
```

5. Configure a loopback interface.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 1.2.3.4/32
```

6. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

7. Configure the overlay gateway.

```
overlay-gateway g1
type layer2-extension
ip interface Loopback 1
  map vni auto
activate
```

Configuring an EVPN Layer 3 VNI

Follow these steps to configure an EVPN Layer 3 VNI.

Layer 3 VNI is used to support VRFs.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 85.
2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
  local-as 100
    neighbor 98.0.0.1 remote-as 100
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  address-family l2vpn evpn
    graceful-restart
    neighbor 98.0.0.1 encapsulation vxlan
    neighbor 98.0.0.1 activate
```

5. Configure a loopback interface.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 1.2.3.4/32
```

6. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

7. Configure the overlay gateway.

```
overlay-gateway g1
type layer2-extension
ip interface Loopback 1
  map vni auto
activate
```

8. Configure a VRF.

```
vrf red
rd 5:50
evpn irb ve 100 <peer-gateway> <--Identifies the L3-VNI
address-family ipv4 unicast
  route-target export 5:100 evpn
  route-target import 5:100 evpn
!
address-family ipv6 unicast
  route-target export 5:100 evpn
  route-target import 5:100 evpn
```

9. Configure the L3-VNI Integrated Routing and Bridging (IRB) instance. (An IP address is not required.)

```
vlan 100  <-- The L3-VNI is a VLAN as a result of auto map mode
router-interface Ve 100
```

10. Alternatively, configure the L3-VNI IRB by using a BD. (The L3-VNI is 4K+BD-ID as a result of auto mapping.)

```
bridge-domain 500 p2mp
 router-interface ve 100
!
interface Ve 100
vrf forwarding red
no shutdown
```

Configuring an EVPN LVTEP for MAC IP routes

Follow these steps to configure an EVPN LVTEP for MAC IP routes.

This configuration is similar to that for Layer 2 EVPN MAC routes.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 85.
2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
 local-as 100
  neighbor 3.3.3.3 remote-as 101 <-- The VXLAN peer
  neighbor 40.40.100.50 remote-as 102 <-- The MCT peer
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 address-family l2vpn evpn
  graceful-restart
  neighbor 3.3.3.3 encapsulation vxlan <-- The VXLAN peer
  neighbor 3.3.3.3 activate
  neighbor 40.40.100.50 encapsulation mpls <-- The MCT peer
  neighbor 40.40.100.50 activate
```

5. Configure a loopback interface for BGP neighborship.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 40.40.100.40/32
```

6. Configure a loopback interface for VXLAN. (MCT peers must have the same address.)

```
interface Loopback 2
 no shutdown
 ip ospf area 0
 ip address 2.2.2.2/32
```

7. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

8. Configure the overlay gateway.

```
overlay-gateway g1
 type layer2-extension
 ip interface Loopback 1
  map vni auto
 activate
```

9. Configure the MCT cluster.

```
cluster c1 1
 peer-interface Ve 45
 peer 40.40.100.50
 deploy
```

Example show and clear commands for VXLAN Layer 3 gateway

This section presents example output of the **show** and **clear** commands that are useful in managing VXLAN Layer 3 gateway.

show overlay-gateway

The following is example output from the **show overlay-gateway** command.

```
device# show overlay-gateway
Overlay Gateway "VXLAN", ID 1, rbridge-ids 1
Admin state up
IP address 33.32.31.13 (loopback 1), Vrf default-vrf
Number of tunnels 6
Packet count: RX 17909 TX 1247
Byte count : RX (500125) TX 356626
```

show tunnel brief

The following is example output from the **show tunnel brief** command.

```
device# show tunnel brief
Tunnel 1, mode VXLAN, rbridge-ids 1
Admin state up, Oper state up
Source IP 33.32.31.13, Vrf default-vrf
Destination IP 33.32.31.10
Tunnel 2, mode VXLAN, rbridge-ids 1
Admin state up, Oper state up
Source IP 33.32.31.13, Vrf default-vrf
Destination IP 33.32.31.1
```

show tunnel

The following is example output from the **show tunnel** *ID* command.

```
device# show tunnel 1
Tunnel 1, mode VXLAN, rbridge-ids 1
Ifindex 2080374798, Admin state up, Oper state up
Overlay gateway "VXLAN", ID 1
Source IP 33.32.31.13 (loopback 1), Vrf default-vrf
Destination IP 33.32.31.10
Active next hop on rbridge1:
IP: 33.32.31.10, Vrf: default-vrf
Egress L3 port: Ve 10, Outer SMAC: 0027.f886.bb36
Outer DMAC: e41f.1343.97d2
Egress L2 Port: Po 11, Outer ctag: 10
BUM forwarder: yes
Packet count: RX 18492 TX 1242
Byte count : RX (NA) TX 356307
```

show vlan brief

The following is example output from the **show vlan brief** command, to verify VNI mapping.

```
device# show vlan brief
Total Number of VLANs configured : 1
Total Number of VLANs provisioned : 1
Total Number of VLANs unprovisioned : 0
VLAN Name State Ports Classification
(F)-FCoE (u)-Untagged, (t)-Tagged
(R)-RSPAN (c)-Converged
(T)-TRANSPARENT
30 VLAN0030 ACTIVE Po 11(t)
Tu 1(t) vni 100
Tu 2(t) vni 100
```

show mac-address-table

The following is example output from the **show mac-address-table** command with the **bridge-domain** keyword, to verify gateway MAC addresses.

```
device# show mac-address-table bridge-domain
VlanId/BD-Id   Mac-address           Type     State     Ports/LIF/peer-ip
629(B)            0011.2222.5555     Dynamic  Active    eth 1/3.100
629(B)            0011.2222.6666     Dynamic  Inactive  eth 1/1.500
629(B)            0011.2222.1122     Dynamic  Active    10.12.12.12
629(B)            0011.2222.3333     static   Inactive  po 5.700
629(B)            0011.0101.5555     Dynamic  Active    eth 1/2.400

Total MAC addresses : 5
```

show ip arp suppression-cache

The following is example output from the **show ip arp suppression-cache** command, to verify remote ARPs and NDs..

```
device# show ip arp suppression-cache
Flags: L  - Locally Learnt Adjacency
       R  - Remote Learnt Adjacency
       RS - Remote Static Adjacency
Vlan/Bd    IP              Mac             Interface                       Age           Flags
-------------------------------------------------------------------------------------------------
0100 (V)   10.10.10.11     0000.0a0a.0a0b X/X                          00:01:35        L
0101 (V)   10.10.10.98     0000.1111.0000 X/X                          Never           RS
0101 (V)   10.10.10.99     609c.9f5a.4d15 X/X                          Never           RS
0102 (V)   12.12.122.1     609c.9f5a.4715 X/X                          Never           RS
```

show bgp evpn

All the options under the **show bgp evpn** command are helptul.

The following is example output from the **show bgp evpn l3vni** command, for a specific VRF .

```
device# show bgp evpn l3vni vrf red

------------------------------------------------------------------------
   L3VNI Prefix Origination Conditions for vrf (red)
------------------------------------------------------------------------
Address Family under BGP : True
RD Configured            : True
IRB I/F Configured        : True (0x48000064)
IRB I/F Status            : False
IRB EVID Conigured        : True (100)
Router mac Exists        : True
Source VTEP              : 40.40.40.1
VTEP Active              : Active
IPv4 L3VNI Active             : Active
IPv6 L3VNI Active             : Inactive


------------------------------------------------------------------------
   L3VNI Prefix Import Conditions for vrf (red)
------------------------------------------------------------------------
Address Family under BGP : True
IRB I/F Configured       : True (0x48000064)
IRB EVID Configured      : True (100)
Router mac Exists        : True
IPv4 L3VNI Active             : Active
IPv6 L3VNI Active             : Inactive
```

The following is example output from the **show bgp evpn routes** command, with ARP specified.

```
device# show bgp evpn routes type arp

Total number of BGP EVPN ARP Routes : 4 Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP
D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
Route Distinguisher: 40.40.100.50:32869
1      ARP:[0][0000.0a0a.0a0b]:[IPv4:10.10.10.11]
                          0.0.0.0         0          100        0      BL
         AS_PATH:
            L2 Label: 101 L3 Label: 100
         ESI : 00.000000000000000000
2      ARP:[0][609c.9f5a.4715]:[IPv4:15.143.15.1]
                          0.0.0.0         0          100        0      BL
         AS_PATH:
            L2 Label: 101 L3 Label: 100
         ESI : 00.000000000000000000
Route Distinguisher: 40.40.100.50:33769
3      ARP:[0][609c.9f5a.4715]:[IPv4:14.13.15.1]
                          0.0.0.0         0          100        0      BL
         AS_PATH:
            L2 Label: 1001 L3 Label: 100
         ESI : 00.000000000000000000
Route Distinguisher: 40.40.100.60:32869
4      ARP:[0][609c.9f5a.8d15]:[IPv4:6.6.2.5]
                          40.40.40.2      0          100        0      BE
         AS_PATH: 1000
            L2 Label: 101 L3 Label: 0
         ESI : 00.000000000000000000
```

The following is example output from the **show bgp evpn routes** command, with IPv4 prefixes specified.

```
device# show bgp evpn routes type ipv4-prefix briief
Total number of BGP EVPN Ipv4Prefix Routes : 4 Status codes: s suppressed, d damped, h history, * valid, >
```

```
best, i internal, S stale Origin codes: i - IGP, e - EGP, ? - incomplete
    Network            Next Hop        MED        LocPrf     Weight Path
Route Distinguisher: 5:50
*>  IP4Prefix:[0][14.13.15.0/24]
                       0.0.0.0         0          100        0      ?
*>  IP4Prefix:[0][15.143.15.0/24]
                       0.0.0.0         0          100        0      ?
*>  IP4Prefix:[0][16.16.16.0/24]
                       0.0.0.0         0          100        0      ?
Route Distinguisher: 5:100
*>  IP4Prefix:[0][17.17.17.0/24]
                       40.40.40.2      0          100        0      1000 ?
```

show tunnel statistics

The following is example output from the **show tunnel statistics** command.

```
device# show tunnel statistics
Tnl ID RXpackets TX packets RX bytes TX bytes
======== ============== ============== ============== ================
1      18573       1242       356307     356307
```

clear counters all

Use the **clear counters all** command to clear statistics on a gateway, including tunnel statistics.

clear overlay-gateway

Use the **clear overlay-gateway** command to clear statistics on an overlay gateway, including tunnel statistics.

BD VE support for VXLAN Layer 3 gateway

BD VE overview

An SLX device can act as a VXLAN Layer 3 gateway (L3GW) to connect different network segments. With L3GW, the device routes VXLAN Layer 2 and Layer 3 traffic over a VXLAN tunnel and conversely.

For Layer 3 forwarding, a virtual Etrhernet (VE) IP interface is configured under a bridge domain (BD). This VE interface can be part of a default or nondefault VRF. No routing protocols (such as OSPFor IS-IS) are supported over such a VE, which is a regular connected subnet. However, this subnet can be announced by routing protocols when it is configured on other IP interfaces in the device. L3GW is supported between SLX 9850 series platforms, or between those platforms and VDX 6740 series platforms that act as L3GW routers.

> NOTE
> While L3GW can be configured between SLX 9850 series platforms and NSX Hypervisor virtual machines (VMs), this configuration is not supported.

Feature components

The following components support this feature.

BD

A bridge domain is a Layer 2 broadcast domain that provides for the forwarding of VXLAN data packets. VXLAN Network Identifiers (VNIs) that identify VXLAN networks (VNs) must be mapped to BDs in 1:1 mode, so that a BD can function as a VXLAN network entity to transmit VXLAN traffic.

BD interface

A BD interface (BDIF) is a Layer 3 logical interface that is created for a BD. Configuring IP addresses for BDIF interfaces allows communication between VXLANs on different network segments and between VXLANs and non-VXLANs, and implements Layer 2 network access to a Layer 3 network.

VNI

A VNI is a VXLAN segment identifier similar to a VLAN ID. VMs on different VXLAN segments cannot communicate directly at Layer 2. A VNI identifies only one tenant. Even if multiple terminal users belong to the same VNI, they are considered as a single tenant. A VNI consists of 24 bits and supports a maximum of 16 M tenants. In distributed VXLAN gateway scenarios, a VNI can be a Layer 2 or a Layer 3 VNI.

A Layer 2 VNI is mapped to a BD in 1:1 mode for the intrasegment transmission of VXLAN packets. A Layer 3 VNI is bound to a VPN.

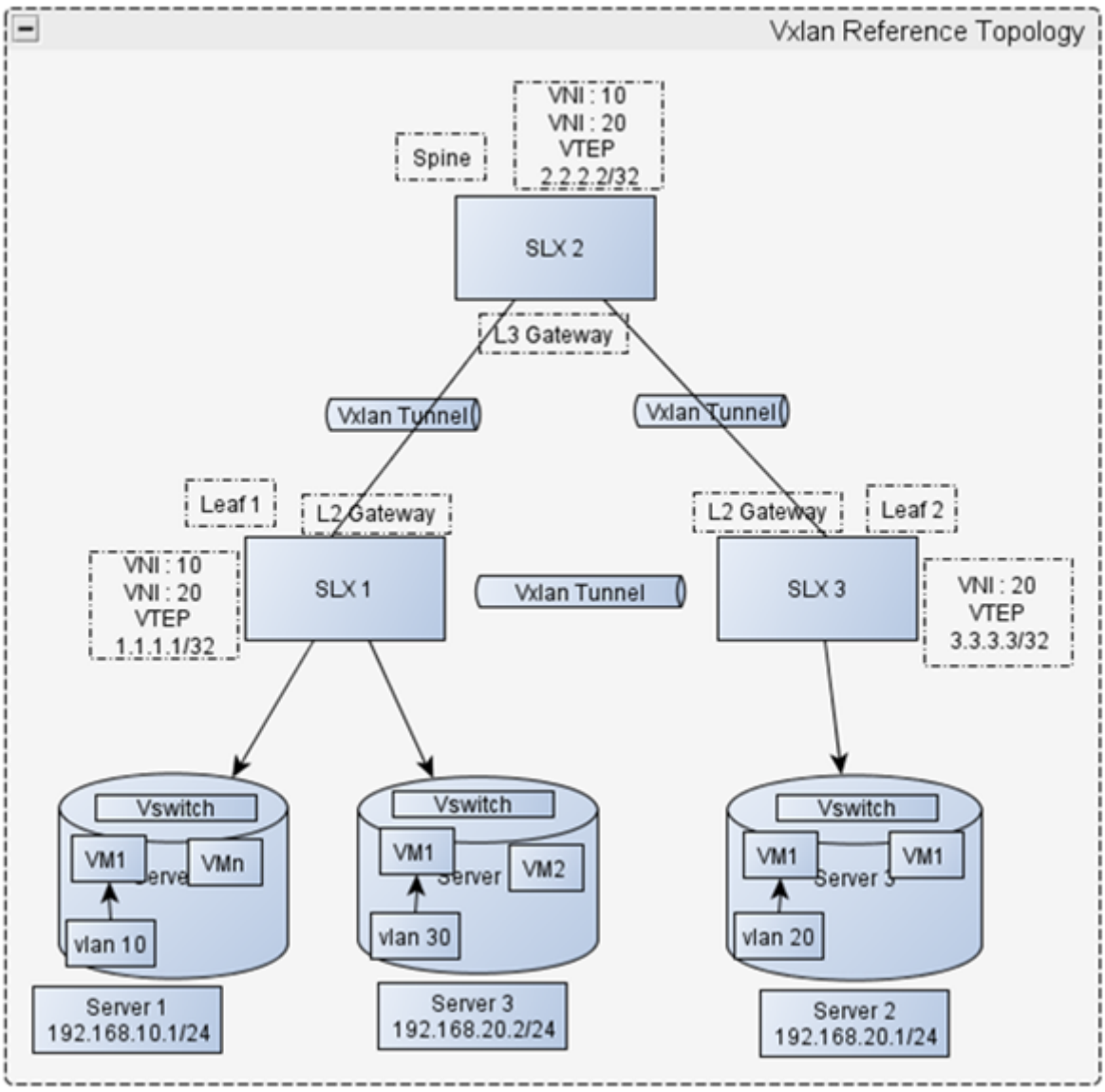
A VXLAN tunnel is identified by a pair of VXLAN tunnel endpoint (VTEP) IP addresses. A VXLAN tunnel is statically created after the user configures local and remote VNIs and VTEP IP addresses, and the tunnel goes up when the pair of VTEPs are reachable at Layer 3.

Topologies

The following topologies illustrate L3GW centralized and distributed scenarios.

The following figure illustrates a centralized VXLAN gateway scenario.

FIGURE 11 Centralized VXLAN gateway scenario

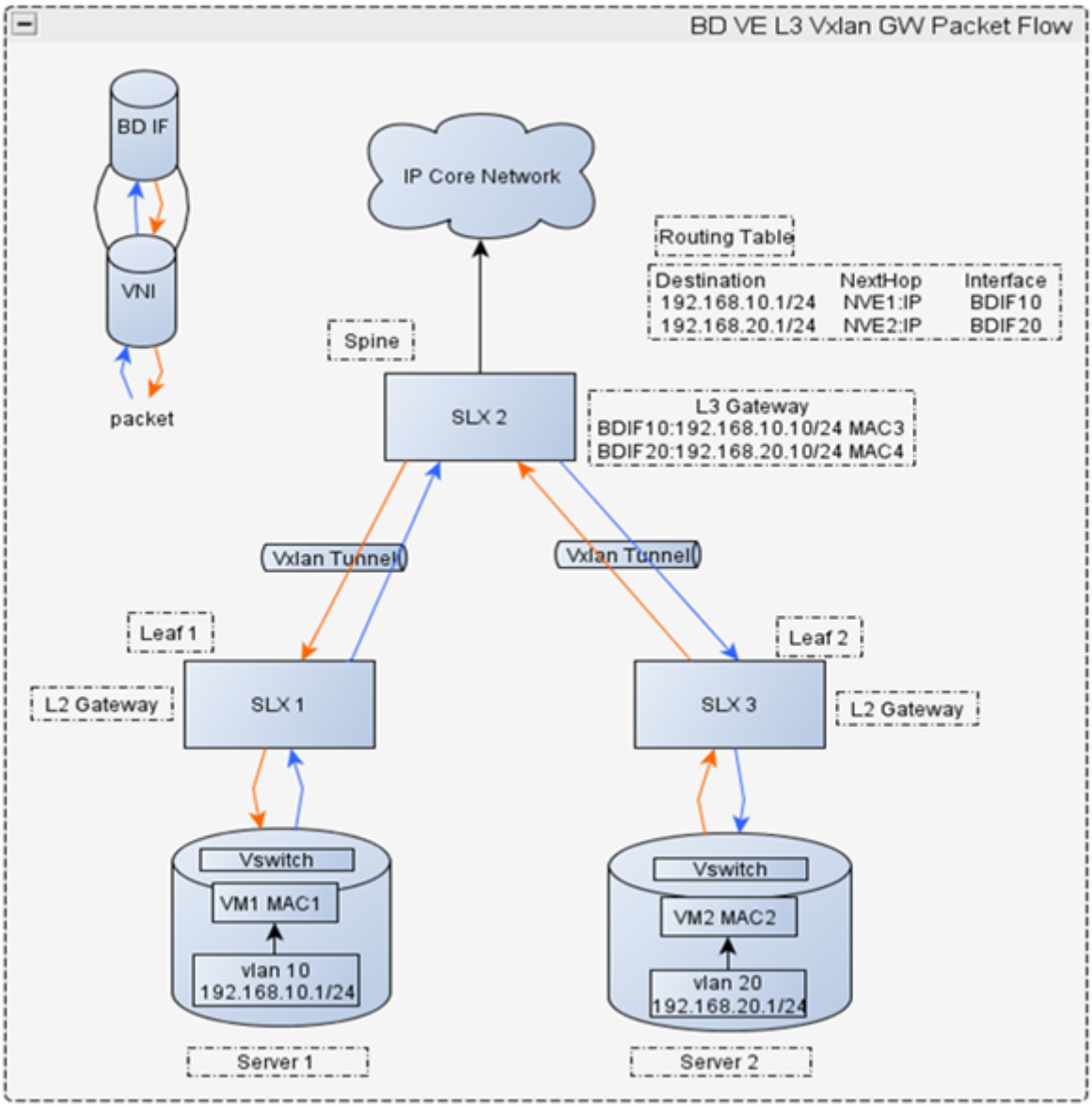


In the network shown above, Server 1 and Server 3 are deployed for SLX 1, and Server 2 is deployed for SLX 3. Server 1 and Server 2 reside on different network segments, whereas Server 2 and Server 3 reside on the same network segment. To allow VMs on Server 2 and Server 3 to communicate, VNIs and VTEP IP addresses must be configured to establish a VXLAN tunnel between SLX 1 and SLX 3. To allow VMs on Server 1 and Server 2 to communicate, VNIs and VTEP IP addresses must be configured to establish a VXLAN tunnel between SLX 1 and SLX 2 as well as between SLX 2 and SLX 3.

The IP interface connected to Server 1 on SLX 1 and the IP interface connected to Server 2 can be configured on a user-defined VRF.

The following figure illustrates a scenario for distributed VXLAN gateway packet forwarding.

FIGURE 12 Distributed VXLAN gateway packet forwarding



In the network shown above, the intersegment packet forwarding process is as follows:

1. After SLX 2 receives VXLAN packets, it decapsulates them and checks whether the destination MAC address in the inner packets is the MAC address of the Layer 3 gateway interface, BDIF10. If the destination MAC address is a local MAC address, SLX 2 forwards the packets to the Layer 3 gateway on the destination network segment and proceeds to Step 2. If the destination MAC address is not a local MAC address, Device 3 searches for the outbound interface and encapsulation information in the Layer 2 BD.
2. SLX 2 removes the Ethernet headers of the inner packets and parses the destination IP address. SLX 2 searches the routing table for the next-hop IP address (based on the destination) and the ARP entries (based on the next-hop IP address). SLX 2 uses the ARP entries to identify the destination MAC address, the outbound interface of the VXLAN tunnel, and the VNI. If the

outbound interface of the VXLAN tunnel and the VNI cannot be found, SLX 2 performs Layer 3 forwarding. If the VXLAN tunnel's outbound interface and VNI can be found, SLX 2 proceeds to Step 3.

3. SLX 2 encapsulates the VXLAN packets again, with the source MAC address in the Ethernet header of the inner packets as the MAC address of the Layer 3 gateway interface, BDIF20.

Layer 3 support: limitations and considerations

This section lists the limitations and considerations for this feature.

The IP next-hop for the underlay tunnel must not be through the subnet that is configured for the VE interface over the tunnel. However, this is not restricted in software.

The following table lists supported and unsupported features for VXLAN underlay.

TABLE 18 Supported and unsupported features for VXLAN underlay

Supported	Not supported
Both nondefault and default VRF configurations	Proxy ARP
ARP learning on the tunnel	L3 protocols (not enforced)
Static routes	RPF (not enforced)
Connected subnets, which can be redistributed over other protocols	VRRP/e (not enforced)
Pinging of connected subnet hosts over a VXLAN tunnel	L2 or L3 multicast (not enforced)
Traceroute	PDUs that exceed MTU size (if the packet to be sent over the VXLAN tunnel exceeds MTU size, the packet is dropped)
	Keepalive of any kind
	High availability or ISSU

The following table lists supported and unsupported features for routing.

TABLE 19 Supported and unsupported features for routing

Supported	Not supported
IP routing from a VE-over-VXLAN tunnel to a pure VE and conversely	IP routing between any other tunnel type (such as GRE/IP tunnel, MPLS-based pseudowire tunnels) to a VXLAN tunnel (not enforced; such configurations must be avoided)
IP routing from one VE-over-VXLAN tunnel to another	Inter-VRF

BD ID range

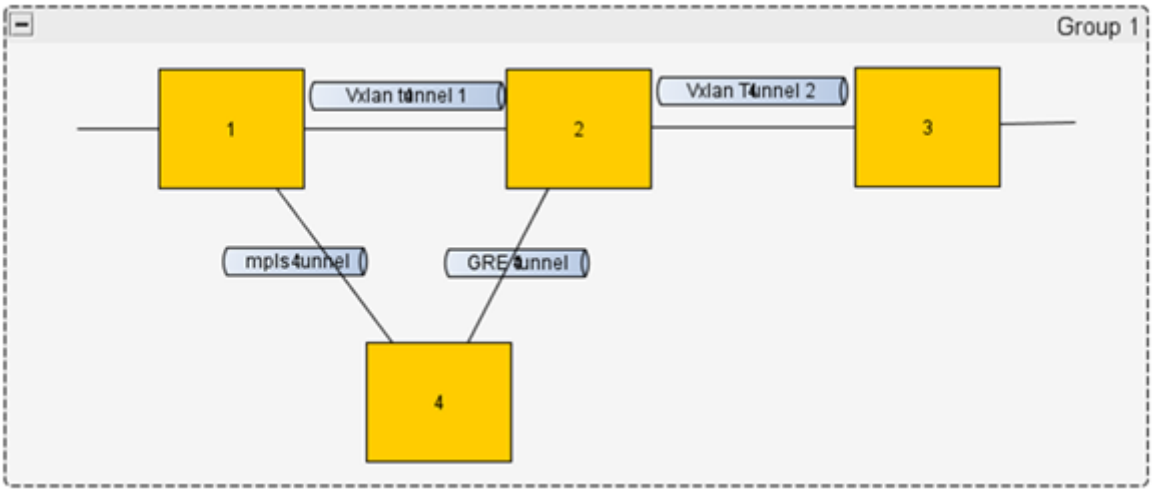
User-configured BD ID values range from 1 through 4 K.

Advanced routing scenarios

The following figure illustrates a topology that supports routing where the following scenarios are allowed:

- Routing between one VXLAN tunnel and another
- Routing between a VXLAN tunnel and an MPLS tunnel and vice-versa
- Routing between any other tunnel type (such as GRE/IP) and a VXLAN tunnel and vice-versa

FIGURE 13 Advanced routing scenarios



Configuring BD VE support for VXLAN Layer 3 gateway

Configuring a static VXLAN tunnel

Follow these steps to configure a static VXLAN tunnel.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 85.
2. Enter global configuration mode.

   ```
   device# configure terminal
   ```

3. Enter the **overlay-gateway** command and specify the name of a gateway.

   ```
   device(config)# overlay-gateway l3vxlangw
   ```

4. Enter the **type** command and confirm that Layer 2 extension is specified.

   ```
   device(config-overlay-gw-l3vxlangw)# type l2-extension
   ```

   Layer 2 extension is the only type supported.
5. Enter the **ip interface loopback** command and specify a loopback interface.

   ```
   device(config-overlay-gw-l3vxlangw)# ip interface loopback 1
   ```

   This specifies the loopback port number for the overlay gateway instance.
6. Enter the **map bridge-domain** command and map a bridge domain to a VNI.

   ```
   device(config-overlay-gw-l3vxlangw)# map bridge-domain 100 to vni 10000
   ```

7. Enter the **site** command and specify a site.

   ```
   device(config-overlay-gw-l3vxlangw)# site vtep1
   ```

8. Enter the **ip address** command and specify an IPv4 address for the site.

```
device(config-overlay-gw-l3vxlangw-site-vtep1)# ip address 1.1.1.1
```

9. Enter the **extend bridge-domain** command to add the VLAN, and exit to overlay-gateway configuration mode.

```
device(config-overlay-gw-l3vxlangw-site-vtep1)# extend bridge-domain add 100
device(config-overlay-gw-l3vxlangw)# exit
```

10. Enter the **activate** command to enable the VXLAN gateway.

```
device(config-overlay-gw-l3vxlangw)# activate
```

Example static VXLAN tunnel configurations

This section presents static VXLAN tunnel configurations on two peer SLX nodes.

Node A configurations

```
vrf red
rd 1:1
address-family ipv4 unicast
  ip route 92.92.92.0/24 11.0.1.1

bridge-domain 100 p2mp   <--Mapped BD router-interface Ve 100
pw-profile default
 bpdu-drop-enable
 local-switching

overlay-gateway l3vxlangw   <--Gateway configuration
type layer2-extension
ip interface Loopback 1
map bridge-domain 100 vni 10000
activate
site vtep1
ip address 1.1.1.1
extend bridge-domain add 100

interface Ve 100   <--Mapped BD interface
 vrf forwarding red
 ip proxy-arp
 ip address 11.0.1.2/24
 shutdown

interface Ethernet 0/5   <--Access interface
 vrf forwarding red
 ip proxy-arp
 ip address 91.91.91.1/24
 no shutdown

interface Ethernet 0/3   <--VXLAN tunnel interface
 ip proxy-arp
 ip address 10.0.1.2/24
 no shutdown

interface Loopback 1
 ip address 2.2.2.2/32
 no shutdown

ip route 92.92.92.2/32 11.0.1.1   <--Default VRF static route
ip route 10.10.10.0/24 next-hop-vrf red 9.9.9.2   <--Nondefault VRF static route
```

Node B configurations

```
vrf red
rd 1:1
address-family ipv4 unicast
  ip route 92.92.92.0/24 11.0.1.1

bridge-domain 100 p2mp   <--Mapped BD
 router-interface Ve 100
pw-profile default
 bpdu-drop-enable
 local-switching

overlay-gateway l3vxlangw   <--Gateway configuration
type layer2-extension
ip interface Loopback 1
map bridge-domain 100 vni 10000
activate
site vtep1
ip address 2.2.2.2
extend bridge-domain add 100

interface Ve 100   <--Mapped BD interface
 vrf forwarding red
 ip proxy-arp
 ip address 11.0.1.1/24
 shutdown

interface Ethernet 0/4   <--Access interface
 vrf forwarding red
 ip proxy-arp
 ip address 92.92.92.1/24
 no shutdown

interface Ethernet 0/3   <--VXLAN tunnel interface
 ip proxy-arp
 ip address 10.0.1.1/24
 no shutdown

interface Loopback 1
 ip address 1.1.1.1/32
 no shutdown

ip route 92.92.92.1/32 11.0.1.2   <--Static route
```

NOTE

Use the **map vni auto** command to map a BD ID to a VNI automatically.

Example BGP EVPN-based configurations

BGP EVPN configuration is required for automatic VXLAN tunnel discovery and the automatic mapping of VLANs/BDs to VNIs.

The following examples illustrate the configuration of peering interfaces, BGP, an overlay gateway, and EVPN.

Peering interfaces

The following example shows how to create a VLAN/BD and a VE interface and add the VLAN to the physical port connecting R1 and R2.

R1

```
vlan 100
 router-interface Ve 100
!

interface Ve 100
```

```
 ip ospf area 0
 ip proxy-arp
 ip address 10.0.0.1/24
 no shutdown
!

interface Ethernet 0/1   <--Change this according to the physical connection
 switchport
 switchport mode trunk
 switchport trunk allowed vlan all
 no shutdown
!
```

R2

```
vlan 100
 router-interface Ve 100
!
interface Ve 100
 ip ospf area 0
 ip proxy-arp
 ip address 10.0.0.2/24
 no shutdown
```

BGP

The following example shows how to configure a local AS and a remote neighbor IP address and activate the neighbor under EVPN address-family.

R1

```
router bgp
 local-as 100
 neighbor 10.0.0.2 remote-as 100
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 address-family evpn
  neighbor 10.0.0.2 activate
 !
!
```

R2

```
router bgp
 local-as 100
 neighbor 10.0.0.1 remote-as 100
 address-family ipv4 unicast
 !
 address-family evpn
  neighbor 10.0.0.1 activate
 !
!
```

Overlay gateway

The following example shows how to configure a source VTEP loopback interface and an overlay gateway.

R1

```
interface Loopback 1
 ip address 1.2.3.4/32
 no shutdown
!

overlay-gateway g1
```

```
 type layer2-extension
 ip interface Loopback 1
 map vni auto
 activate
!
```

R2

```
interface Loopback 1
 ip address 10.20.30.40/32
 no shutdown
!

overlay-gateway g1
 type layer2-extension
 ip interface Loopback 1
 map vni auto
 activate
!
```

EVPN

The following example shows how to configure EVPN.

```
evpn l3gwevpn
 rd auto
 bridge-domain add 100
```

BD VE show commands

This section presents example output of a variety of **show** commands that are useful in BD VE configuration.

show bridge-domain

```
device# show bridge-domain 100
Bridge-domain 100
------------------------------
Bridge-domain Type: MP, VC-ID: 10000 MCT Enabled: FALSE
Number of configured end-points:  1, Number of Active end-points: 1
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
MAC Withdrawal: Disabled
PW-profile: default, mac-limit: 0
Total VPLS peers: 0 (0 Operational):
```

show tunnel

```
device# show tunnel 61441
Tunnel 61441, mode VXLAN, node-ids 1
Ifindex 0x7c00f001, Admin state up, Oper state up
Overlay gateway "l3vxlangw", ID 1
Source IP 1.1.1.1 (  Loopback 1 ), Vrf default-vrf
Destination IP 2.2.2.2
Configuration source Site
MAC learning enabled
Active next hops on node 1:
    IP: 10.0.1.2, Vrf: default-vrf
    Egress L3 port: Eth 0/5, Outer SMAC: 609c.9f5a.5d1d
    Outer DMAC: 609c.9f5a.551b, ctag: 0
    BUM forwarder: yes

Packet count: RX 2594261         TX 52
Byte count  : RX 306122333       TX 6783
```

show tunnel brief

```
device# show tunnel brief
Tunnel 61441, mode VXLAN, node-ids 1
Admin state up, Oper state up
Source IP 1.1.1.1 (  Loopback 1 ), Vrf default-vrf
Destination IP 2.2.2.2
```

show tunnel status

```
device# show tunnel status
Tnl id    Adm state Oper state BFD state Tnl dest IP     VeId(GRE) Ve state(GRE)
========= ========= ========== ========= =============== ========= ==============
61441     up        up                  100.11.11.11     NA         NA
SLX# show tunnel statistics
Tnl ID   RX packets      TX packets      RX bytes             TX bytes
======== =============== =============== ==================== ====================
61441    2699275         52              318513985            6783
```

show arp

```
device# show arp
Entries in VRF default-vrf : 3
Address          Mac-address     Interface     MacResolved   Age         Type
-----------------------------------------------------------------------------
10.0.1.2         609c.9f5a.551b  Eth 0/5       yes           00:51:47    Dynamic
11.0.1.2         609c.9f5a.5515  Ve 100        yes           00:00:43    Dynamic
92.92.92.2       0000.0ac3.c9ad  Eth 0/4       yes           00:42:12    Dynamic
```

show mac

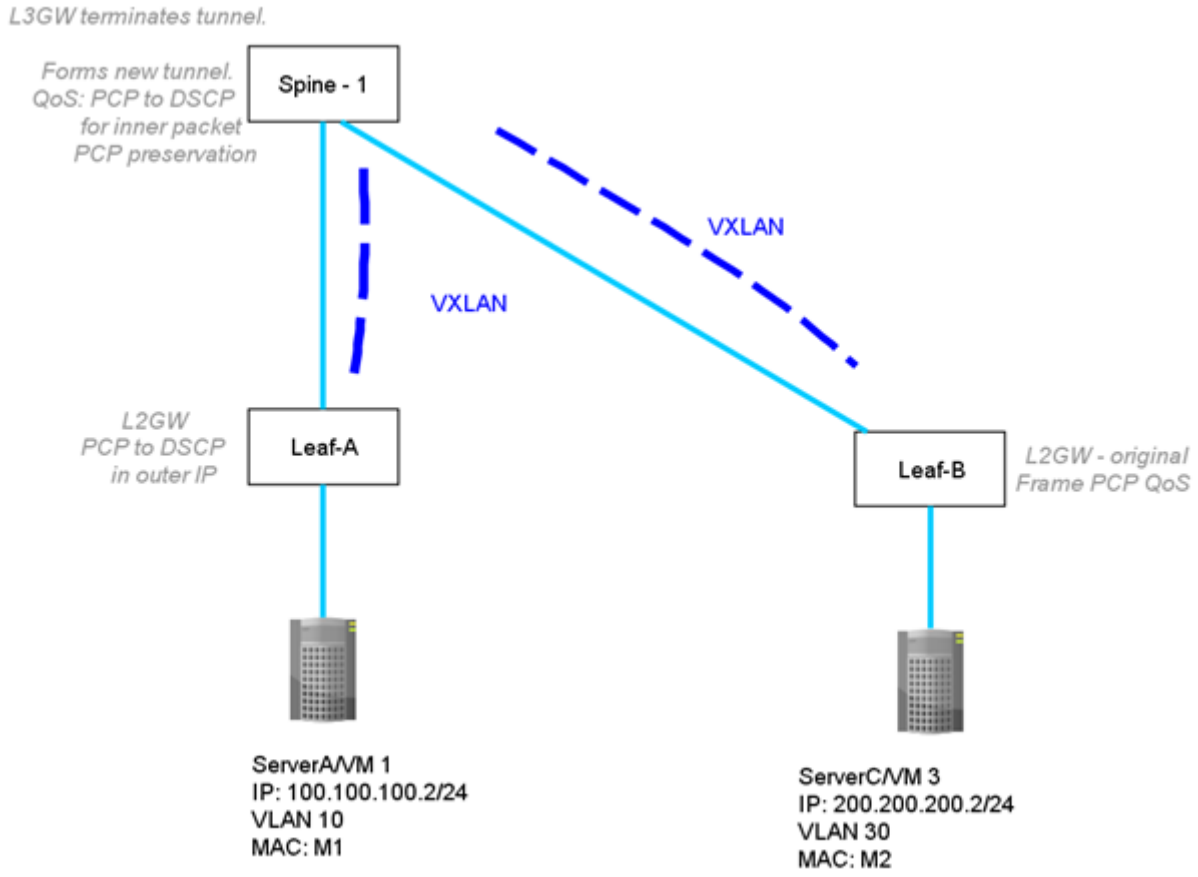
```
device# show mac
VlanId/BDId   Mac-address       Type        State      Ports/LIF/PW
100 (B)       609c.9f5a.5515    Dynamic     Active     tu61441
Total MAC addresses    :  1
```

QoS for VXLAN Layer 2 and Layer 3 gateway interconnections

SLX-OS VXLAN Layer 2 and Layer 3 gateway interconnections can support QoS.

VXLAN Layer 3 gateways resolve the ARP for host VMs and create routes. When VM1 sends intersubnet packets to VM3, Leaf-A sends the packet to Spine - 1 over the VXLAN configuration in the following figure. First, Spine - 1 de-encapsulates the original L2 frame inside the VXLAN tunnel, and the L2 frame has DMAC as the gateway MAC of Spine - 1. Then, Spine - 1 routes the packet to VM3, where the ARP resolution is through the VXLAN configuration.

FIGURE 14 QoS for VXLAN Layer 2 and Layer 3 gateway interconnections



Configuring QoS for VXLAN Layer 2 and Layer 3 gateway interconnections

Enter the **qos-ttl-mode uniform** command to set the QoS type mode to uniform, which is the default, when configuring the VXLAN L2 gateway. For example, enter the following commands when configuring the gateway for the ingress (tunneling) side of packet forwarding:

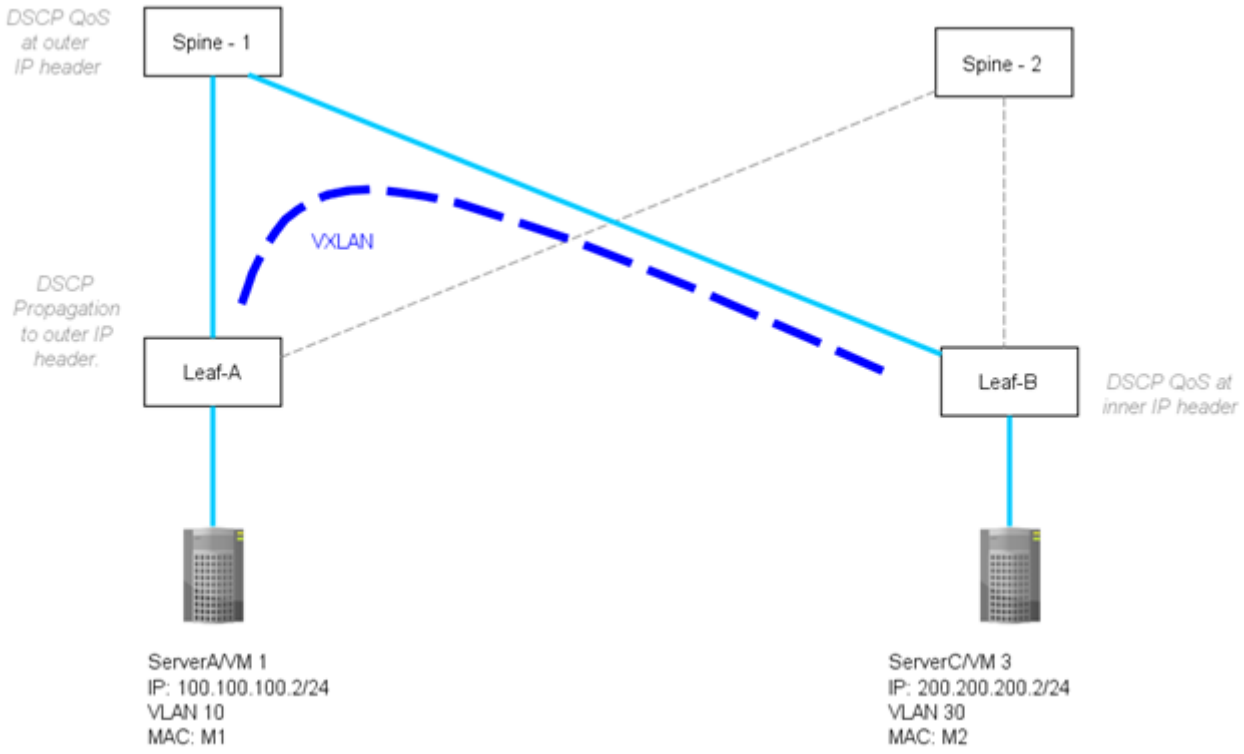
```
device(config)# overlay-gateway gateway_L2L3
device(config-overlay-gw-gateway_L2L3)# type layer2-extension
device(config-overlay-gw-gateway_L2L3)# ip interface loopback 1
device(config-overlay-gw-gateway_L2L3)# qos-ttl-mode uniform
device(config-overlay-gw-gateway_L2L3)# map vni auto
device(config-overlay-gw-gateway_L2L3)# activate
```

QoS for VXLAN Layer 3 gateways

SLX-OS VXLAN Layer 3 gateways can support QoS.

A VXLAN L3 gateway allows intersubnet communication through the VXLAN configuration. In the following figure, the leaf nodes act as the VXLAN L3 gateway while Spine - 1 and Spine - 2 act as IP routing nodes that just route the VXLAN packet out between Leaf-A and Leaf-B. Hosts on different subnets cannot learn MAC addresses of each other in a VXLAN network. The leaf nodes resolve ARP for a tenant VM. The leaf nodes advertise the host routes to each other and tenant VMs with the next hop configured as VTEP. In the following figure, VM1 on a VXLAN network is sending a packet to VM3 on a VXLAN network. At Leaf-A, the DSCP of the tenant packet is propagated to the outer IP header. Leaf-A also adds a L2 header corresponding to the VTEP gateway MAC, after the VXLAN header. Optionally, the 802.1p VLAN/PCP marking of this inner Ethernet header may be set to a value corresponding to the received Layer 2 traffic class of the tenant packet. The Leaf-B terminating IP and L2 headers ignore the QoS marking of transport network and proceed with marking the inner IP header.

FIGURE 15 VXLAN L3 gateway support of intersubnet communication



Configuring QoS for VXLAN Layer 3 gateways

First, complete the required configuration for VXLAN Layer 3 gateways, refer to Configuring TCAM profiles to support Layer 3 gateway on page 85. Then, enter the **qos-ttl-mode uniform** command to set the QoS type mode to uniform, which is the default, when configuring the VXLAN L3 gateway:

```
device(config)# overlay-gateway gateway_L3
device(config-overlay-gw-gateway_L3)# type layer2-extension
device(config-overlay-gw-gateway_L3)# ip interface loopback 1
device(config-overlay-gw-gateway_L3)# qos-ttl-mode uniform
device(config-overlay-gw-gateway_L3)# map vni auto
device(config-overlay-gw-gateway_L3)# activate
```

Multi-Chassis Trunking (MCT)

MCT Overview

Multi-Chassis Trunking (MCT) is trunking that initiates at a single MCT-unaware server or switch and terminates at two MCT-aware switches. MCT allows the links to the two MCT-aware switches to appear to a downstream device as if they are coming from a single device on a single Link Aggregation (LAG) trunk interface or physical port.

NOTE
The SLX-OS device does not support Layer 2 protocols over MCT. You must provide a loop-free topology.

NOTE
MCT does not support any variant of Spanning Tree Protocol (STP). STP is disabled by default and must not be enabled with MCT.

In a datacenter network environment, LAG trunks provide link level redundancy and increased capacity. However, they do not provide switch-level redundancy. If the switch connected to the LAG trunk fails, the entire trunk loses network connectivity.

With MCT, member links of the LAG trunk are connected to two MCT-aware devices. A configuration between the devices enable data flow and control messages between them to establish a logical Inter-Chassis Link (ICL). In this model, if one MCT device fails, a data path remains through the other device.

SLX-OS Layer 2 MCT is based on RFC 7432 (BGP Ethernet VPN). The MP-BGP EVPN extension is the control plane on the SLX-OS device to perform both MAC synchronization and cluster management. MAC synchronization between the MCT peers synchronizes the MAC tables between the MCT nodes for node resiliency and faster convergence.

For the data plane, the SLX-OS device supports the MPLS forwarding mechanism to leverage MPLS LER functionality.

SLX-OS MCT provides Layer 3 protocol support for IPv4 and IPv6 BGP, OSPF, and IS-IS through a VLAN or bridge domain VE interface. The VE over MCT interface MAC address is synchronized to the MCT peer through BGP by using an EVPN MAC route.

SLX-OS supports MCT over VPLS or VLL Layer 2 VPN services, Multicast, or IPv4 or IPv6 Virtual Routing Redundancy Protocol (VRRP) and VRRP Extended (VRRP-E). For more information on Multicast over MCT, see the *Extreme SLX-OS IP Multicast Configuration Guide*.

MCT terminology

Before implementing MCT in your network, you must understand some key terms and definitions.

MCT peer devices	A pair of SLX-OS device configured as peers. The LAG interface is spread across two MCT peer devices and acts as the single logical endpoint to the MCT client. NOTE MCT is supported across the same chassis type only; for example, SLX-9850 <---> SLX-9850.
MCT client	The MCT client is the device that connects with the MCT peer devices. It can be a switch or an endpoint server host in the single-level MCT topology or another pair of MCT switches in a multi-tier MCT topology.
MP-BGP EVPN extension	The control plane for Layer 2 MCT on the SLX-OS device.
MCT Cluster Client Edge Port (CCEP)	A port on one of the MCT peer devices that is a member of the LAG interface to the MCT client. To have a running MCT instance, at least one Link Aggregation Interface is needed with a member port on each peer device. While there is a LAG on the client device, CCEP on the MCT device can be a LAG or a physical port.
MCT Cluster Edge Port (CEP)	A port on MCT peer device that a member of a MCT VLAN and is not a Cluster Client Edge Port.
MCT VLANs	VLANs on which MCT clients are operating. These VLANs are explicitly configured in the MCT configuration.

SLX-OS MCT control plane

Multiprotocol-BGP (MP-BGP) EVPN extension, as specified in RFC 7432, is used as the SLX-OS MCT control plane.

The control plane consist of the following components:

- EVPN instance—Mapped to a Layer 2 VLAN that the RFC refers to the VLAN-Based service interface.
- Ethernet segment ID (ESI)—10-byte integer that uniquely identifies the set of links connecting MCT PEs to the client CE. SLX-OS MCT supports both dynamic and static LAG between MCT PE and CE and uses ESI type 0 that is encoded as follows:
  - 1-byte ESI type = 0
  - 9-byte ESI value = user input through the SLX-OS **esi** command

  When the client interface is a port channel and LACP is running on the port channel, MCT supports an automatically generated ESI value, as defined in RFC 7432. This ESI is encoded as type 1, as follows:
  - 1-byte ESI type = 1
  - 9-byte ESI value = 6-byte LACP system MAC address of the client followed by the 2-byte LACP port key, and then a 1-byte 0x00
- MP-BGP route distinguisher (RD)—Encoded using RD type 1 as defined in RFC 4364 that consists of the following subfields:
  - 4-byte administrator subfield that is set with the 4-byte router ID
  - 2-byte assigned number subfield that is encoded with the all zeros for the Ethernet segment (ES) route, client ID for the Ethernet Auto-Discovery route, and EVPN ID (VLAN ID) for MAC and multicast routes.
- MP-BGP EVPN capability—When a BGP session is brought up to a MCT peer, BGP indicates to the peer that it is EVPN capable using BGP capability advertisement with the following information:
  - Capability code = 1 (MP-BGP)
  - AFI = 25 (L2VPN)
  - SAFI = 70 (EVPN)

  If a BGP session already existed to the same peer, the existing BGP session is flapped to allow the advertisement of the EVPN capability.
- MP-BGP EVPN route types—Includes Ethernet Auto-Discovery (A-D), MAC/IP Advertisement, Inclusive Multicast Ethernet Tag, and Ethernet Segment routes.

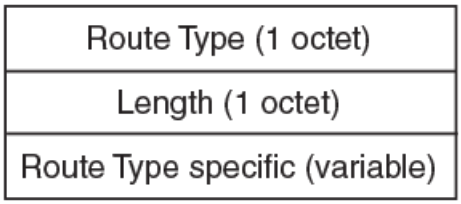
- MPLS Label Assignment—Statically assigned label ranges for ESI label, EVPN unicast label and EVPN BUM label.
  - The ESI label is generated based on the start ESI label value plus the client ID.
  - EVPN unicast or BUM label is generated based on the start unicast or BUM label plus the VLAN ID.
- Designated forwarder—A PE in a set of multi-homing PEs connected to the same Ethernet segment that is elected for sending BUM traffic to a client for a VLAN ID on an Ethernet segment.

SLX-OS MCT operates in dual-homing mode.

MP-BGP EVPN Routes

RFC 7432 defines EVPN Network Layer Reachability Information (NLRI) with the format as shown in the following figure:

FIGURE 16 EVPN NLRI format



SLX-OS MCT supports the following route types.

TABLE 20 SLX-OS MCT route types

Route type	Route name	SLX-OS usage
1	Ethernet Auto-Discovery (per Ethernet Segment only)	Mass MAC withdraw and Designated forwarder election. The PE advertises one Ethernet A-D route for each client interface. When the client interface goes down, the PE withdraws the Ethernet A-D route which is served as a trigger for the remote PE to remove all MAC addresses learned over the affected client interface instead of withdrawing an individual MAC.
2	MAC/IP Advertisement	MAC synchronization of the MAC addresses between two MCT peers.
3	Inclusive Multicast Ethernet Tag	Advertisement of ingress replication usage and multicast label expected for each EVPN instance when receiving BUM traffic.
4	Ethernet Segment (ES)	Designated forwarder election. The ES route is used to update the remote peer when the MCT client is deployed and undeployed.

Designated forwarder election

Designated forwarder election is triggered in the following scenarios:

- A client is deployed locally or remotely.
- BGP session comes up.
- CCEP goes up or down.

To elect a designated forwarder (DF) for a VLAN ID on an Ethernet segment, each PE exchanges its router IP address with its multi-homing PEs through the Ethernet Segment route. The following algorithm uses the IP address to select the DF.

1. Upon the discovery of a new ES, a PE advertises the ES route and waits a default of three seconds for its peer to advertise the ES route.

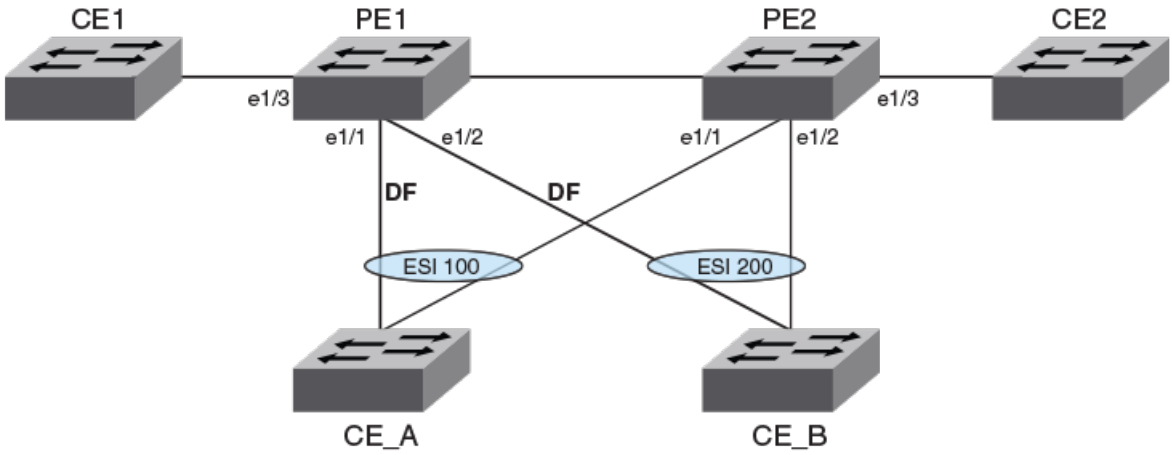
2. When the timer expires, the PE builds a sorted list of PE IP addresses including its own address connected to the same ES.
3. The PE with the ordinal number that equals (V mod N) is elected the DF. V is the VLAN ID and N is the number of PEs.
4. When the ES link fails, the PE withdraws its ES route which triggers the selection process to select a new DF. When a PE node failure occurs, DF election is also triggered when the PE is up and down.

   NOTE
   DF election is triggered only after the client is deployed on both MCT devices.

SLX-OS MCT data plane traffic

For the discussion of the MCT data plane traffic, refer to the following topology diagram and configuration.

FIGURE 17 EVPN MPLS forwarding topology



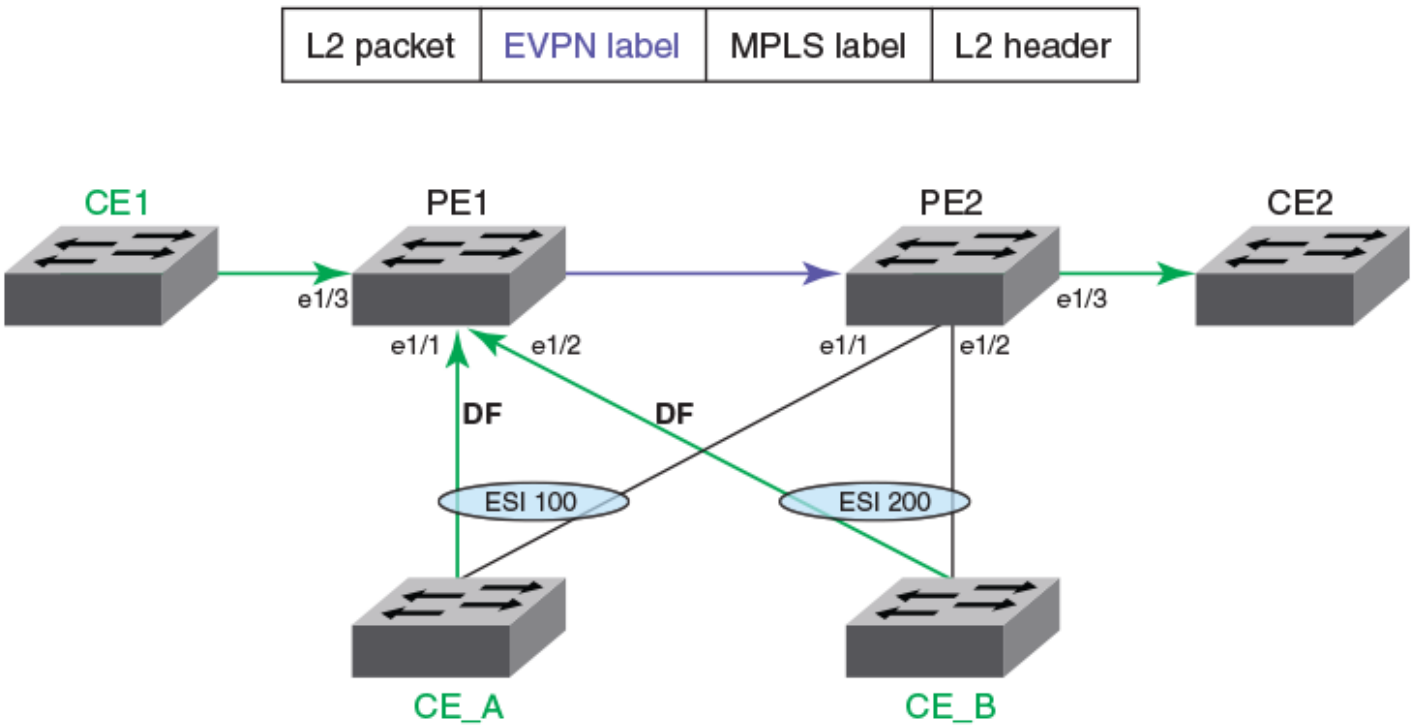
- Cluster VLAN 1000 is configured on both PE1 and PE2.
- Cluster VLAN 1000 has client interface e1/1 and e1/2 that belong to ESI 100 and 200 respectively and a cluster end point on e1/3.
- PE1 is elected as Designated Forwarder (DF) for VLAN 1000 for both ESI 100 and 200.
- VLAN 1000 is mapped to VSI/EVPN instance 1000.

Unicast traffic sent between CE1 and CE_A or CE_B follows normal L2 forwarding.

Forwarding unicast traffic between PEs

To send unicast traffic received over a CEP or CCEPs on PE1 to the remote PE2, PE1 pushes an previously-received PE2 EVPN label. The following figure shows the packet encapsulation.

FIGURE 18 EVPN unicast forwarding between cluster PEs



For an SLX-OS device, it advertises one label for all the MACs learned within a bridge domain.

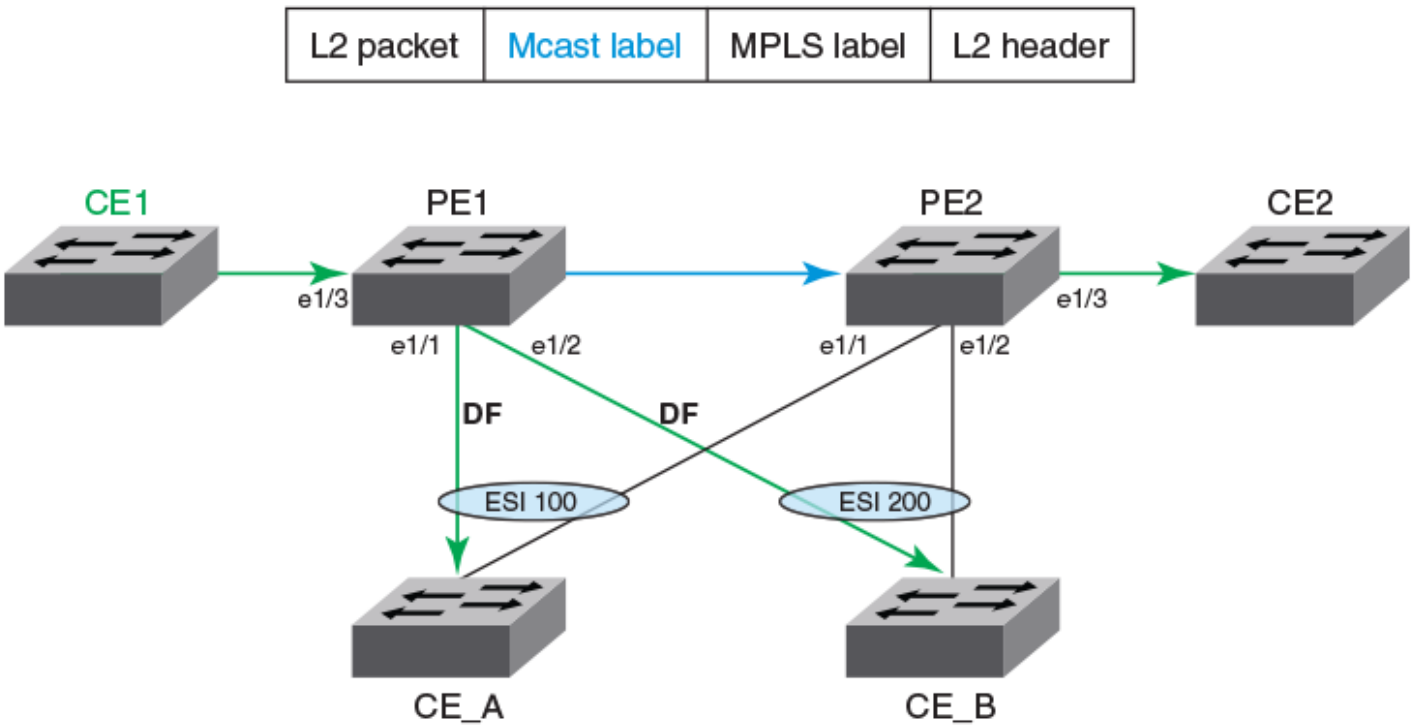
NOTE

This traffic forwarding is also applicable for MCT VPLS and MCT VLL. For MCT VLL, there are no CEP ports. There is no EVPN label. Only the MPLS label is involved for communication between cluster peers. For each bridge domain, the forwarding rules are defined based on the DF role.

Flooding traffic received from a CEP

For unicast traffic received over a CEP that PE1 needs to flood on VLAN 1000, PE1 sends a copy to e1/1, e1/2, and to PE2. The copy sent on e1/1 and e1/2 has basic L2 encapsulation as in the case of a normal L2 VLAN. The copy sent to PE2 has a multicast label previously received from PE2 encapsulated as shown in the following figure.

FIGURE 19 EVPN BUM forwarding between cluster PEs

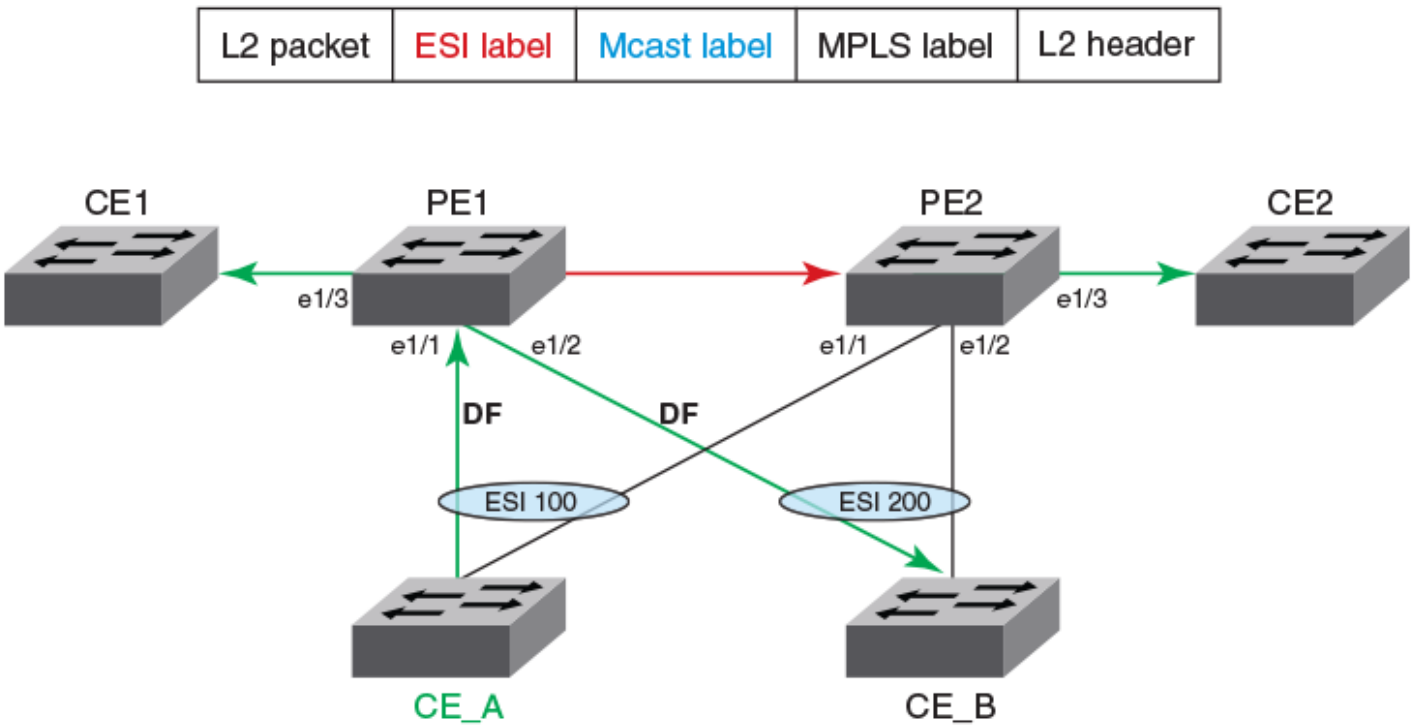


For SLX-OS device, it advertises one multicast label for each EVPN instance.

Flooding traffic received from CCEP

For traffic received over a CCEP from CE_A on PE1 to be flooded to remote PE2, PE1 must push previously received multicast and ESI labels from PE2. The following figure shows the packet encapsulation.

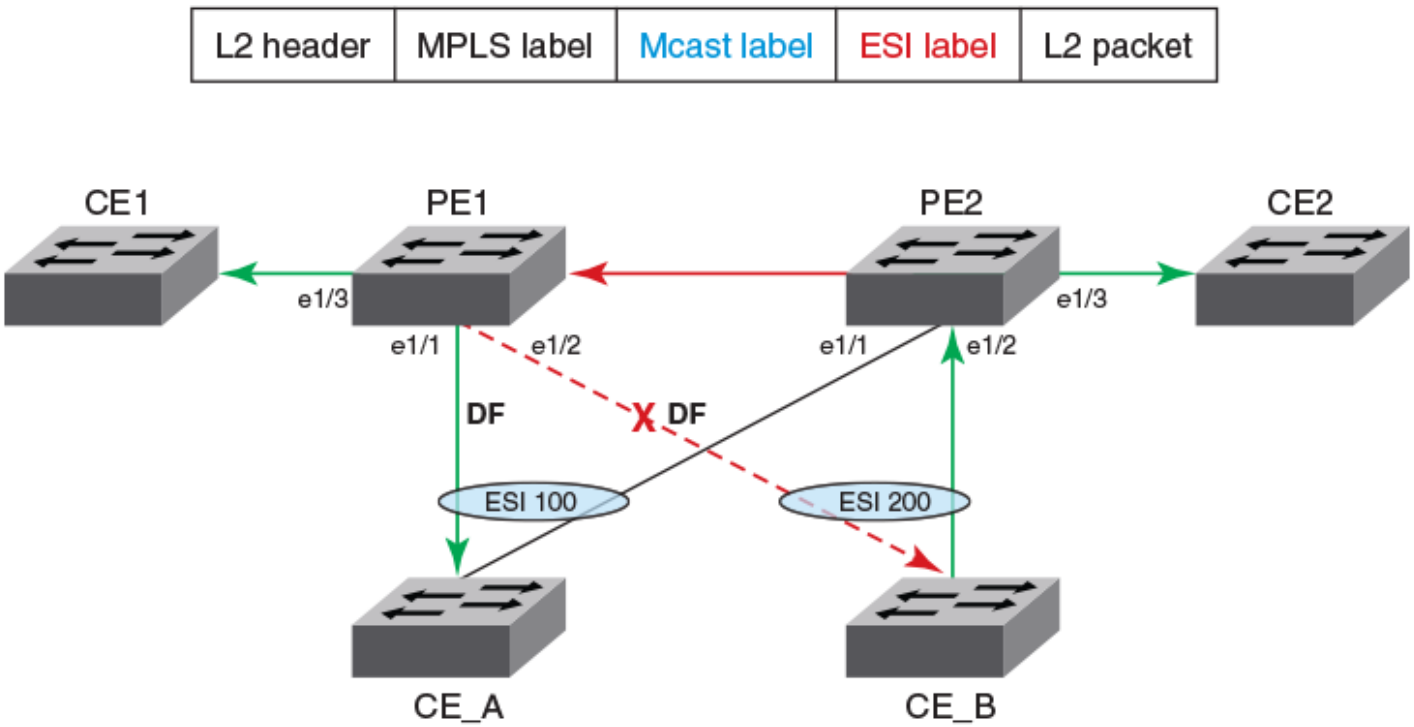
FIGURE 20 EVPN source port suppression based on DF election



Since source port suppression is based on the DF election result, PE2 suppresses copies to be sent to CE_A and CE_B because e1/1 and e1/2 are not elected as DF for ESI 100 and 200 respectively.

In the following figure, source port suppression is done based on the ESI label carried in the packet itself.

FIGURE 21 EVPN source port suppression based on the ESI label



For traffic received from CE_B on PE2 that is flooded to PE1, PE1 sends a copy to CE1 and CE_A on e1/1. However, it suppresses a copy to be sent to CE_B. The suppression is done because the copy received from PE2 carried an ESI label that PE1 previously advertised to PE2 for ESI 200.

MAC management

Each MAC entry maintains information about all the owners of the MAC entry who learned and advertised it. The information about each owner is maintained in the form of the MAC database (MDB).

A MDB entry contains peer and client information with the cost. A local MAC entry has cost 0 and MAC addresses learned from an MCT peer has a cost of 1. The MDB entry with the lowest cost is chosen as the filtering database (FDB).

In MCT topologies, MAC addresses can be learned either on local CCEP or CEP interfaces, or from the remote MCT node. If the same MAC is learned from both MCT nodes, then MAC entries learned locally have higher priority than the one learned from the remote peer.

For remote MAC addresses, aging is disabled, and they can only be deleted when delete notifications are received from the remote node that advertised it before for learning.

The following terminologies are associated with MCT MAC entries.

- Dynamic—MAC addresses learned locally as EVPN on CEP ports
- EVPN—MAC addresses learned on remote CEP ports
- Cluster Client Local (CCL)—MAC addresses learned locally on a client interface
- Cluster Client Remote (CCR)—MAC addresses learned on a remote client interface

Static MAC handling

Configuration of static MAC entries is allowed over MCT enabled VLANs and CEP and CCEP interfaces.

The MCT static MAC addresses configured on a local node are advertised to remote MCT node for learning. While advertising the MAC using the BGP MAC advertisement route, it uses the MAC mobility extended-community route to identify the MAC as static using the sticky MAC field. On the remote node, when MAC advertisement is received for a static MAC address, the sticky MAC information is saved along with the MAC entry.

When an MCT static MAC address is deleted, a BGP MAC withdrawal route is sent to the remote peer to delete the MAC entry from its database.

When a CEP interface is down and if any static MAC entries are present, MAC Delete messages are sent to the remote node to flush the entries.

When a CCEP interface is down and if any static MAC entries are associated with the client, the MAC addresses are moved to point to the remote MCT peer. The MAC addresses are moved back to the CCEP when the interface comes back up.

On a local MCT node, when a cluster is UP and you configure a static MAC on a CEP or CCEP interface, the node synchronizes the MAC address to the remote MCT node. The remote node processes the MAC address and adds it to the FDB. On the remote MCT node, you can configure the same MAC as the static MAC address for the client 1 CCEP interface since it is configured on the same client CCEP interface. No additional static MAC configurations on the remote node are required since the same MAC are already part of the local MCT node.

When the cluster is down on the local and remote MCT nodes, both nodes are independent as clusters that can be independently configured with the static MAC addresses for the CEP or CCEP interface. However, when the cluster is brought up, the static MAC addresses are synchronized from both nodes and the addresses on the remote node are rejected since the local configuration takes precedence. The misconfiguration remains until you correct it.

MAC learning

MAC learning over CEP interfaces is like basic Layer 2 learning and the EVPN MAC advertisement route is sent to the cluster peer to synchronize the learned MAC address. MAC learning over CCEP interfaces is two-step process in which the MAC entry is added into the MDB first. The best MAC entry is chosen and installed into the FDB and the EVPN MAC advertisement route is sent to the cluster peer for synchronization of the learned MAC address.

The following rules are used for MAC learning:

- If a static MAC address is configured on the CEP port, it is learned as the Static and the EVPN MAC advertisement route message is sent to the peer. In this case, the ESI is set to NULL. On the peer MCT node, the EVPN MAC address programmed on the cluster peer is static towards the MCT peer.
- If a static MAC address is configured on the CCEP port, an EVPN MAC advertisement route is sent to the peer. In this case, the MAC entry is associated with the ESI of the MCT client. The peer MCT node programs the MAC address as static over the local CCEP interface.
- Dynamic MAC learning from the CEP is similar to basic Layer 2 MAC learning. An EVPN MAC advertisement route is sent to the peer. In this case, the ESI is set to invalid or NULL. On the peer MCT node, the EVPN MAC address programmed on the cluster peer is static towards the MCT peer.
- Dynamic MAC learning from the CCEP occurs as a CCL MAC. An EVPN MAC advertisement route is sent to the peer. In this case, the MAC entry is associated with the ESI of the cluster client. The peer MCT node programs the MAC address as static over the local CCEP interface.

MAC aging rules

The following rules are defined for MAC aging:

- The local MAC age over CEP interface is similar to the Layer 2 MAC age. After the local MAC delete, an EVPN MAC withdrawal route is sent to the MCT peer.
- The local MAC age over CCEP interface is considered aged only if all MCT nodes age out the entry. When the MAC that ages on one of the MCT node local MDB is deleted, if the remote MDB present MAC is reprogrammed as the CCR, else the MAC is removed from the local FDB, an EVPN MAC withdrawal route is sent to MCT peer.
- The remote MAC addresses of the EVPN and CCR that are learned through the EVPN MAC advertisement route does not age out. They can only be removed by the EVPN MAC withdraw messages from the peer.

MAC movement

A MAC address is considered to be moved when the same MAC address is received on a different interface with same VLAN. In MCT, a MAC movement is allowed on both local and remote nodes.

The following table describes the allowed MAC movements in MCT.

TABLE 21 MCT MAC movement

MAC movement scenario	Behavior
Local dynamic MAC move from CEP1 to the CEP2 edge interface on MCT1.	On local node MCT1, the MAC address is updated to point to the new CEP2 interface. There is no MAC route update required to the remote MCT node. As on the remote node, the MAC always point towards the MCT peer for all EVPN MAC addresses.
Local dynamic MAC move from CEP1 edge interface to the CCEP1 client interface on MCT1.	On local node MCT1, the MAC address is updated to point to the new client interface CCEP1. A MAC update route is sent with the new ESI of client 1. The remote node updates the MAC address to point to the CCEP of client 1.
CCEP1 interface (client 1) to CCEP2 interface (client 2) on MCT1.	On local node MCT1, the MAC address is updated to point to the new client interface CCEP2. A MAC update is sent with the new ESI of client 2 to the remote node. The remote node updates the MAC address to point to the CCEP of client 2.
Local dynamic MAC move from CCEP1 interface (client 1) to CEP1 edge interface on MCT1.	On local node MCT1, the MAC address is updated to point to the new edge interface CEP1. A MAC update is sent with the new ESI 0 to the remote node. The remote node MCT2 updates the MAC address pointing to the MCT1 node.
For a MAC learned on a CEP port locally (MCT1). Dynamic MAC move to a CEP port on the remote node (MCT2)	On the MCT2 node for the EVPN MAC learned from MCT1, it is considered as a MAC move when it is learned on a CEP port. The MAC is updated as local on MCT2 and now points to the Dynamic on the CEP port on MCT2 instead of pointing to MCT1 node MCT2 sends an updated MAC to MCT1. MCT1 updates the MAC as remote and points to the MCT2 .
For a MAC learned on a CEP port locally (MCT1). Dynamic MAC move to CCEP1 on MCT2.	On the MCT2 node for the EVPN MAC learned from MCT1, it is considered as a MAC move when the same MAC is learned on a CCEP1 port. The MAC is updated as CCL on MCT2 and now points to the local CCEP1 port on MCT2 instead of pointing to the MCT1 node. MCT2 sends a CCL MAC updated to MCT1. MCT1 updates the MAC as CCR and point to the CCEP1 port.
For a MAC CCL learned on a CCEP1 port locally (MCT1). Dynamic MAC move to CCEP2 on remote MCT2 node.	On the MCT2 node for the CCR MAC learned from MCT1 for client 1, it is considered as a MAC move when the same MAC is learned on client 2 over the CCEP2 port. The MAC is updated as CCL on MCT2 and now points to the local CCEP2 port on MCT2 instead of pointing to CCEP1. From MCT2, it sends a CCL MAC updated to MCT1. MCT1 updates the MAC as CCR and points to the CCEP2 port.

TABLE 21 MCT MAC movement (continued)

MAC movement scenario	Behavior
For a MAC CCL learned on a CCEP1 port locally (MCT1). Dynamic MAC move to CEP port on remote MCT2 node.	On the MCT2 node for the CCR MAC learned from MCT1 for client1, it is considered as a MAC move when the same MAC is learned on the CEP port. The MAC is updated as Dynamic on MCT2 and points to the local CEP port on MCT2 instead of pointing to CCEP1. From MCT2, it sends the EVPN MAC updated to MCT1. MCT1 updates the MAC as EVPN and points to the MCT2.

MAC address deletion

The following rules are defined for MAC address deletion. Note that every deletion triggers the MAC resolution algorithm and reprograms the MAC entry if required.

- If the CEP interface is down, MAC addresses are deleted locally and individual MAC deletion messages are sent to the peer.
- If the CCEP local port is down and the remote CCEP is down, MAC addresses are deleted locally and the ESI withdraw message is sent to the MCT peer instead of sending individual MAC delete messages.
- If the CCEP local port is down and the remote CCEP is up, all local MAC addresses are moved to point to the remote MCT peer including the static MAC addresses associated with the CCEP.
- When the client entry is undeployed, all MAC addresses are deleted locally, and the ESI withdraw message is sent to the MCT peer to delete all associated client MAC addresses.

Automatic RSVP LSP bring up for MCT

The automatic bring up of an RSVP LSP for MCT functionality allows MPLS to automatically create and delete an RSVP LSP when you deploy or undeploy the MCT cluster with a peer-interface configuration. This functionality handles MM failover and the MPLS process restart for automatically-created LSPs.

NOTE
This LSP is a non-CSPF LSP.

Automatic LSP creation

When you deploy the MCT cluster, MPLS receives the cluster peer interface and a peer address for the LSP creation. The peer interface can be any Layer 3 interface (physical or VE). The peer address is the peer router ID.

Upon receiving the cluster deploy update, MPLS is enabled on the peer interface and the RSVP LSP is created with a default template with a unique auto-generated name, *MCT_peer-ipaddr_peer-interface-index*. A random number suffix is generated if this name is already in use in the user configurations. You can see the LSP by using the **show mpls lsp** command.

Automatic LSP deletion

When the cluster is undeployed, MCT updates the MPLS daemon to delete the automatically-created RSVP LSP. Then the MPLS daemon deletes the corresponding LSP. MPLS is disabled on the peer interface if it is not MPLS enabled from the CLI configuration.

MCT configuration considerations

General considerations

- When the primary physical interface is also an MCT cluster client interface, logical interfaces that belong to that primary interface cannot be in the same MCT bridge domain (BD).
- MCT does not support any variant of Spanning Tree Protocol (STP) on ICL links, cluster client ports, or any VLAN that is part of the cluster. STP is disabled by default.

  NOTE
  If you enable STP on MCT nodes, each node acts as an independent (not interconnected) STP switch. This situation results in STP state flap at the node connected to the CCEP port, because it receives two different Bridge Protocol Data Units (BPDU) on that CCEP port

- SLX-OS supports dynamic and static LAG between the MCT PE and CE.
- SLX-OS uses Ethernet segment identifier (ESI) type 0 (static LAG) and type 1 (dynamic LAG). Consider the following for ESI type 0:
  - Configure the 9-byte ESI value that is used to form a globally unique 10-byte integer ESI.
  - Configure the same 9-byte ESI value for each client on both MCT devices.
- SLX MCT configurations (Layer 2 or Layer 3) do not require the Advance Feature license.

Peer considerations

- For both MCT peers, the MCT peer address must match the peer's BGP router ID (loopback address).
- You must configure the same client ID on both MCT peers for the link or CCEP that is connected to the same client.
- Layer 3 connectivity should allow a BGP session to be established between MCT peers.
- SLX MCT peers must be physically connected to each other.
- You must configure and activate a BGP EVPN neighbor as the peer interface.
- (Required) Prevent native VLAN traffic from going across the ICL by disabling the default native VLAN on the ICL trunk port between the MCT peers. To prevent native VLAN traffic, use the **switchport mode trunk-no-default-native** command from interface configuration mode. For example:

```
device(conf-if-eth-0/17)# switchport mode trunk-no-default-native
```

- MCT MPLS interfaces are automatically created when you configure MCT without the use of router MPLS. By design, automatically created MCT MPLS interfaces do not offer FRR (fast reroute) protection, because it is difficult to remove the FRR configuration from outside of MPLS.

  NOTE
  You need an Advance Feature license if you want to enable FRR for non-MCT paths

VLAN considerations

- To avoid traffic looping, consider these guidelines:
  - Overlay EVPN VLANs or BDs must not be natively configured on the peer interface (underlay interface).

  - If you use a VE as a peer interface, use the corresponding VLAN for underlay alone and do not extend it into the EVPN domain.
- The peer interface is in a separate control VLAN that is not used for MCT members
- If you configure an MCT port channel for multiple VLAN or VE interfaces, use static routes instead of OSPF ECMP.
  - For example, you have two VE interfaces, VE 10 (an MCT VLAN) and VE 20 (a non-MCT VLAN). When OSPF is configured on the interfaces, the route to the BGP router-id (or MCT) peer address is through the interfaces. If you then configure BGP BFD and BGP EVPN sessions between the MCT peers for faster recovery, BFD flaps continuously. Flapping occurs because OSPF does per-packet load balancing of BGP BFD packets, which are not reassembled properly.
  - As a best practice, configure a static route for MCT peers through the MCT VLAN (VE 10 in this example). A static route has a lower administrative distance than OSPF and places only one route in the routing table.

LACP considerations

- When the client interface is a port channel on which LACP is running, MCT supports an automatically generated ESI value, as defined in RFC 7432.
- To ensure that two MCT peers send the same system ID and key but a different port ID to each client, you must configure the same cluster ID and client ID on both nodes. The system ID is derived by appending a cluster ID to the MCT base system ID, which is a device-defined value.

```
System ID = mct_base_system_id (0180.c200) + cluster_id
```

- The LACP fields have the following settings:

```
Key = MCT_LACP_KEY_BASE (3000) + client_ID

Port ID (16-bit unique value) =
5-bit (slot value) + 8-bit (port value + 3-bit) (MCT position offset)
```

LSP considerations

LSP is automatically created when a cluster is deployed. No user configuration is required.

- The SLX-OS device uses the peer interface to set up the MPLS data path automatically. The LSP is the outgoing peer interface and has an MCT prefix when it is displayed by the **show mpls lsp** command.
- The configured peer interface must be the best next hop to reach the MCT peer. If not, the automatically created LSP fails to come up.
- The automatically created LSP can co-exist with other explicit MPLS configurations. You can configure more LDPs and RSVP LSPs, but not with the same name as the automatically created LSP.
- When co-existence occurs on the device:
  - As part of the MCT cluster configuration, the MPLS daemon automatically creates and signals the RSVP LSP with the default template and a unique auto-generated name.
  - If you create an MCT LSP with the same name as an existing LSP, then the MCT LSP is created with a similar name but with a different number suffix to make it unique. For example: `MCT_20.21.22.23_9876543_123456`.
  - You cannot create an LSP with same name as an existing MCT LSP. The attempted configuration is rejected with an error message.
  - When the router port or the VE interface is MPLS-enabled from the MCT configuration, you cannot disable the router port or the VE interface. The configuration to disable the router port or the VE interface from the CLI is rejected with an error message.

  - You can disable MPLS with the **no router mpls** command. Upon command execution, the automatically created LSP and the MPLS-enabled peer-interface configuration are disabled and re-enabled.
- An automatically created LSP is restricted to using the configured MCT peer interface as its next hop connection to the MCT peer. The IGP shortest path between the two peers must match the configured MCT peer interface. Otherwise, the LSP remains operationally down.
  - For example, suppose there are two paths between the MCT peers, eth 1/1 and eth 1/2, and the node configuration for the two peers is symmetrical. Furthermore, the user has configured eth 1/1 as the MCT peer interface.
  - If the IGP SPF path between the two peers uses eth 1/1 as the next hop link, then the LSP becomes operational and uses eth 1/1 as the next hop interface. If the IGP cost of eth 1/1 increases such that eth 1/2 is selected by the IGP SPF calculation, then the LSP remains operationally down.
  - You can use eth 1/2 as a normal MPLS interface. Do not use eth 1/1 (which is the MCT interface) for normal MPLS purposes.

Configuring the BGP EVPN peer

Create a BGP EVPN address family to configure and activate the cluster EVPN peer. This configuration is associated with the MCT cluster peer configuration.

Before configuring a BGP EVPN peer, ensure you configure a loopback interface.

You also create an EVPN instance to add the bridge domains and VLANs associated with the MCT cluster.

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config-terminal)# router bgp
```

3. Configure the EVPN peer with the autonomous system number (ASN).

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 100
```

4. Configure the EVPN peer through the loopback interface.

```
device(config-bgp-router)# neighbor 10.1.1.1 update-source loopback 1
```

5. Enter EVPN address family configuration mode

```
device(config-bgp-router)# address-family l2vpn evpn
```

6. Activate the EVPN peer.

```
device(config-bgp-evpn)# neighbor 10.1.1.1 activate
```

7. Access global configuration mode.

```
device(config-bgp-router)# exit
```

8. Access EVPN configuration mode.

```
device(config)# evpn myinstance
```

9. Configure the bridge domains.

```
device(config-evpn-myinstance)# bridge-domain add 1-2000,4000-4096
```

   For more information on the bridge domain configuration, see the Bridge Domain chapter.

10. Configure the VLANs.

```
device(config-evpn-myinstance)# vlan add 1-2000
```

11. Enable auto-generation of a route distinguisher (RD) for this EVPN instance.

```
device(config-evpn-myinstance)# rd auto
```

12. Enable the auto-generation of the import and export route-target community attributes and ignore the autonomous system (AS) number.

```
device(config-evpn-myinstance)# route-target both auto ignore-as
```

The following example shows the steps in the previous configuration.

```
device# configure terminal
device(config-terminal)# router bgp
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 100
device(config-bgp-router)# neighbor 10.1.1.1 update-source loopback 1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.1.1 activate
device(config-bgp-router)# exit
device(config)# evpn myinstance
device(config-evpn-myinstance)# bridge-domain add 1-2000,4000-4096
device(config-evpn-myinstance)# vlan add 1-2000
device(config-evpn-myinstance)# rd auto
device(config-evpn-myinstance)# route-target both auto ignore-as
```

Configuring MCT

Configure the local and remote MCT cluster and cluster clients.

Before configuring MCT, ensure that the following configurations exist:

- Layer 3 interface for the cluster peer interface
- VLANs and bridge domains under an EVPN configuration to function as the MCT members
- Port channel for Link Aggregation or an Ethernet interface as a client interface

Perform the following steps.

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Create a cluster on the device.

```
device (config)# cluster MCT1 1
```

3. Configure the peer IP address.

```
device(config-cluster-1)# peer 10.1.1.1
```

   The peer IP address must be the remote peer's router ID. This address corresponds with the neighbor in BGP EVPN address family configuration for the peer.

4. Configure the peer interface.

```
device(config-cluster-1)# peer-interface Ve 10
```

   The peer interface should be a valid Layer 3 interface. You should configure the peer interface before deploying the configuration.

5. Deploy the cluster.

```
device(config-cluster-1)# deploy
```

   The MPLS LSP is created when the cluster is deployed and is removed when the cluster in undeployed.

6. Create the client for the cluster and access cluster client configuration mode.

```
device(config-cluster-1)# client MCT1-client 200
```

   On both MCT nodes, you must configure the same client ID.

7. Configure the interface to the cluster client instance.

```
device(config-cluster-client-200)# client-interface port-channel 3
```

   The port channel specifies the LAG ID.

   The client interface can also be a physical interface, for example:

```
device(config-cluster-client-200)# client-interface Ethernet 2/5
```

   The client interface cannot be added under multiple client entries.

8. Set the 9-octet Ethernet Segment ID (ESI) value which is used to uniquely identify the cluster client.

```
device(config-cluster-client-200)# esi 00.a1.b2.c3.d4.e5.f6.89.00
```

   You must configure the same value on both MCT nodes to create the MCT client LAG.

   The ESI cannot be added under multiple client entries.

9. Deploy the cluster client.

```
device(config-cluster-client-200)# deploy
```

10. After configuring the local MCT cluster and client, configure the remote MCT cluster and client.

The following example is the steps in the previous configuration.

```
device# configure terminal
device (config)# cluster MCT1 1
device(config-cluster-1)# peer 10.1.1.1
device(config-cluster-1)# peer-interface Ve 10
device(config-cluster-1)# deploy
device(config-cluster-1)# client MCT1-client 200
device(config-cluster-client-200)# client-interface port-channel 3
device(config-cluster-client-200)# esi 00.a1.b2.c3.d4.e5.f6.89.00
device(config-cluster-client-200)# deploy
```

Taking the MCT node offline for maintenance

If you need to take an MCT device offline for maintenance or an upgrade, perform the following steps to minimize traffic loss.

1. Verify that the MCT node that is peer to the node being taken offline is in loose client-isolation mode.

```
device# show cluster 1
Cluster MCT1 1
==============
Cluster State: Deploy
Client Interfaces Shutdown: FALSE
Client Isolation Mode: Strict
Configured Member Vlan Range: 2, 4-7
Active Member Vlan Range: 2, 4-7
...
```

2. If the peer node is in strict client-isolation mode, configure it to loose mode.

```
device(config)# cluster MCT1 1
device(config-cluster-1)# client-isolation loose
```

3. Disable the MCT clients from the MCT node that you will take offline, as shown in the following example.

```
device(config)# cluster MCT1 1
device(config-cluster-1)# client-interfaces-shutdown
```

4. Isolate the MCT node that you will take offline from the core of the network by shutting down all uplink interfaces.

   NOTE
   Do not write the configuration changes made in the previous steps to the startup-configuration file.

To bring the MCT node back online, perform one of the following actions.

- If you upgraded or downgraded the device, select the **coldboot** option under the firmware download menu.
- For any other reason, execute the **reload system** command. Since the changed configuration was not saved, the reload reverts the configuration changes that had taken the MCT node offline.

Configuring additional MCT cluster parameters

The SLX-OS device has additional cluster commands with default values. You can change the parameters for these commands in cluster configuration mode.

Changing the client-isolation mode

Isolation mode defines the action to be taken when the BGP control session goes down between the MCT nodes while the cluster is in deployed state. When the client-isolation mode is strict, the client interface will be shutdown.

By default, client-isolation mode is based on the peer IP address. The node with the lower peer IP address is set to the client-isolation mode of loose, while the node with the higher peer IP address is set to the client-isolation mode of strict. You can override this behavior by configuring strict client-isolation mode on both nodes.

Use the **client-isolation strict** command to configure the strict mode on both nodes, as shown in the following example.

```
device(config-cluster-1)# client-isolation strict
```

Use the **client-isolation loose** command to configure the loose mode on both nodes, as shown in the following example.

```
device(config-cluster-1)# client-isolation loose
```

Changing the designated-forwarder hold timer value

Upon expiration of the designated-forwarder hold timer, the reelection of the designated forwarder is considered.

By default, the hold time is three seconds. Use the **designated-forwarder-hold-time** command to change the time in seconds from 1 to 60 seconds, as shown in the following example.

```
device(config-cluster-1)# designated-forwarder-hold-time 35
```

Moving the traffic from an MCT node to the remote node

Use the **client-interfaces-shutdown** command to move all the traffic on the node to the remote MCT node by disabling the local client interfaces administratively, as shown in the following example.

```
device(config-cluster-1)# client-interfaces-shutdown
```

Configuring an auto-generated ESI for a cluster client

When a client interface is a port channel and LACP is running on the port channel, you can configure an automatically generated ESI value, as defined in RFC 7432 on the cluster client.

The following example shows how to configure an auto-generated ESI for the cluster client.

```
device(config)# cluster MCT1 1
device(config-cluster-1)# client MCT1-client 200
device(config-cluster-client-200)# esi auto lacp
```

Displaying MCT information

You can display detailed MCT information and related MCT MAC addresses.

To display the EVPN neighbor information, use **show ip bgp neighbors** command. This information includes the peer configured for the EVPN address family, the undeplyed MCT cluster, and the negotiation of the EVPN address family.

Displaying the cluster information

The following example shows the information of the cluster on the SLX-OS device.

```
device# show cluster 1

Cluster MCT1 1
==============
Cluster State: Deploy
Client Interfaces Shutdown: FALSE
Client Isolation Mode: Loose
Configured Member Vlan Range: 2, 4-7
Active Member Vlan Range: 2, 4-7

Peer Info:
----------
Peer IP: 10.1.1.1, State: Up
```

```
  Peer Interface: Ve 20
ICL Tunnel Type: MPLS, State: Up

PW client Info:
---------------

Client Info:
------------
Name     Id   ESI                           Interface      Local/Remote State
access1  100  0:11:22:33:80:0:0:0:0:0        Ethernet 1/8   Up/UP
access2  200  0:11:22:33:81:0:0:0:0:0        Port-Channel 3 Dep/UnDep
access3  300  0:11:22:33:82:0:0:0:0:0        Eth 2/3        Up/Down
```

NOTE

When you delete an IP router ID that is used as the neighbor ID and IP address on an MCT peer, the **show cluster** command on the MCT peer devices displays inconsistent cluster states.

Displaying the cluster client information

The following example displays all client information for cluster 1.

```
device# show cluster 1 client
Client Info:
------------
Name      Id    Label(Lo/Re)    Interface         Local/Remote State
access1   100   NA/ 798721      Ethernet 1/8      UnDep/Dep
access2   200   798722/798722   Port-Channel 3    Up/UP
access3   300   798723/798723   Ethernet 2/3      Down/Up
```

The following example displays client 100 information for cluster 10.

```
device# show cluster 10 client 100
Client Info:
------------
Client: access1, client-id: 100, Deployed, State: Up
Interface: Ethernet 1/8
Vlans : 1-10, 100
Elected DF for vlans: 2, 4, 8, 10, 100
```

Displaying member VLAN information

The following example displays the member VLAN information for the cluster.

```
device# show cluster member vlan
VLAN-ID    Mcast-label(Lo/Re)    Unicast-label(Lo/Re)    Forwarding State
   101     817253 / 817253       800869 / 800869              Up
   102     817254 / 817254       800870 / 800870              Up
   103     817255 / 817255       800871 / 800871              Up
   104     817256 / 817256       800872 / 800872              Up
   105     817257 / 817257       800873 / 800873              Up
   106     817258 / 817258       800874 / 800874              Up
```

Displaying and clearing the MAC address table cluster information

The following example shows how to display the MCT cluster information in the MAC address table.

```
device# show mac-address-table cluster 1
Vlan/BD'Id Mac-address     Type        State    Ports
100(V)     0010.a111.aaaa  CCL         Active   ETH3/1
100(V)     0010.a111.aa22  Static-CCL  Active   ETH3/1
100(V)     0010.a111.bbbb  CCR         Active   ETH3/1
200(V)     003d.a111.1111  Dynamic     Active   Eth 1/1
200(V)     003d.a111.1122  Static      Active   Eth 1/1
```

```
200(V)      003d.a111.3333  EVPN        Active   10.2.2.2
200(V)      003d.a111.3322  EVPN-Static Active   10.2.2.2
```

The MAC Type for an MCT cluster displays the following information:

- The MAC address over the CEP AC endpoints or the VPLS PW are learned and treated as EVPN local MAC addresses over a singled-homed edge node. These local addresses are learned as EVPN and displayed as Dynamic. Corresponding MAC addresses synchronized to the remote MCT node are displayed as EVPN. The EVPN MAC addresses are programmed pointing to the MCT PW.
- For the client MAC behavior, MAC addresses are learned as CCL on the local MCT node and CCR on the remote MCT node pointing to the CCEP interface.
- The VPLS MCT MAC on active PWs are learned as Dynamic and the corresponding MAC addresses are learned as EVPN on remote MCT node.
- Static MAC addresses configured on CEP AC endpoints are learned as Static. The corresponding remote MAC addresses are learned as EVPN-Sticky in the remote node.
- For static MAC addresses over client interfaces, Static-CCL and CCR are displayed.

You can also view the MAC entries for a specific client.

Clearing the MCT cluster MAC table entries

You can clear all cluster entries from the MAC address table or the entries for a specified client. The following example clears the MAC entries for client 3 of cluster 1.

```
device# clear mac-address-table cluster 1 client 3
```

Only the local MAC entries are deleted from the current node. Individual MAC withdrawal flush messages are sent through the EVPN. However, BGP still batches multiple routes to the remote node.

When the remote MCT peer receives the MAC withdrawal message, it only deletes the remote MAC entry. To clear MAC addresses on both nodes, you must issue **clear mac-address-table** commands on both MCT nodes.

VPLS and VLL MCT

VPLS and VLL MCT are used for data center interconnection in which SLX-OS MCT acts as a data center gateway to connect to another data center through either the VPLS or VLL WAN connection.

> NOTE
> For more information on VPLS and VLL, refer to the "VPLS and VLL Layer 2 VPN services" chapter.

For VPLS MCT, a point-to-multipoint (p2mp) bridge domain is added to the MCT cluster. For VLL MCT, a point-to-point (p2p) bridge domain is added to the MCT cluster. The VPLS or VLL horizon is added as a pseudowire (PW) client.

VLL MCT supports PW redundancy. At any point of time, one active-active PW path exists to reach the destination. The node on which the PW is active is called the active node. The endpoint traffic coming from the standby node traverses through the MCT PW session to the active node for that instance and the active MCT node takes care of the forwarding to the remote VPLS or VLL peer.

> NOTE
> For SLX-OS, the MCT cluster requires both nodes to be on SLX-OS devices. However, an SLX-OS MCT cluster that connects to a Extreme MLX cluster through VPLS or VLL is supported.

Control plane for VPLS or VLL MCT

As with Layer 2 MCT, VPLS or VLL MCT uses MP-BGP EVPN for the control plane. However, an Ethernet segment ID (ESI) controls all pseudowires (PWs) and is encoded the same as the Layer 2 CCEP ESI. This ESI is called the PW horizon ESI.

The bridge domain is mapped to an EVPN instance. For each BD, the default EVPN ID is the BD ID plus 4,096. A user configured EVPN ID is not supported. The EVPN ID for the VLAN is the VLAN ID.

For VPLS or VLL MCT, the physical and LAG CCEP operate in active/active multi-homing mode. However, the PW operates in active/standby mode.

The designated forwarder (DF) state of the PW ESI represents the active PW node state for a VPLS or VLL instance. The DF election process for the PW ESI is the same as the Layer 2 ESI process. However, for VLL MCT in the following dynamic cases, VLL does not change its role and is driven through the PW horizon client.

- If the local endpoint is down, the remote endpoint is up.
- If the active-active PW is down on the active node, the active-active PW is up on the standby node.

PW redundancy for VLL MCT

PW redundancy for VLL MCT allows the quick failover of traffic to the backup PWs. To support active and standby PWs to remote PEs, the Preferential status bit in the PW status TLV is exchanged to indicate whether the PW forwarding is active or standby.

Status TLV support is enabled through a VLL instance if one of the following is true.

- VLL is configured with two remote peers.
- The VLL endpoint is a MCT client CCEP port.

To support PW redundancy, configure two VLL peers under one VLL instance. One PW is for each VLL peer. Among these PWs, an active-active PW is selected and used for traffic flow to the remote side. An active-active PW is selected based on the local and remote PW redundancy preferences. A remote PW redundancy preference is received by the PW status TLV. When the bit is set, it indicates PW forwarding standby. When the bit is cleared, it indicates PW forwarding active.

PW state in VPLS or VLL MCT

The PW state in VPLS or VLL MCT is controlled by two entities. The MCT module controls its MCT state. The PW remote peer provides its PW redundancy state. Together, they decide the operational (forwarding) state of the PW. The following table shows the PW state decisions.

MCT state	PW remote state	Operational state	PW signaling state
DF	Active	Active	Active
DF	Standby	Standby	Active
Non-DF	Any	Standby	Standby

When the PW is in active operational state, the data plane objects (such as LIF or MGID, or cross-connect for VLL) is created and be programmed into the hardware. When the PW is in standby operational state, the data plane is programmed as if this PW is down.

NOTE

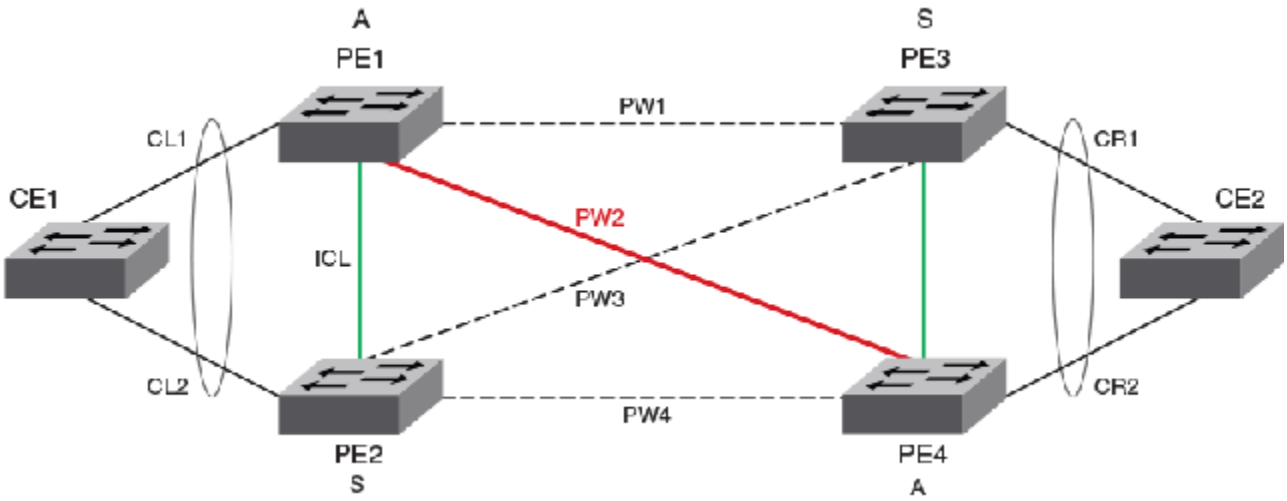
The SLX-OS PW state table is the same as the Extreme NetIron VPLS-MCT or VLL-MCT PW state table to ensure compatibility when facing an MLX MCT cluster over a VPLS or VLL connection.

VLL-MCT data plane

A topology for VLL MCT is provided in the following figure. Two MCT clusters face each other and four PWs connect the clusters.

> NOTE
> This topology is for only one BD. Since the DF state is selected per BD, another BD can use a different PE as the active node.

FIGURE 22 VLL MCT topology



When VLL MCT is activated, only one PW is operationally active between the MCT clusters, as represented by the solid line. The standby PWs are represented by the dotted lines.

The ICL link between the MCT nodes is a BGP EVPN connection. It is not a spoke PW.

> NOTE
> VLL MCT does not use MAC learning. BUM traffic handling is not required. It uses cross-connect instead of VSI. VLL MCT does not use the EVPN label.

Steady state traffic

Based on the previous figure, the steady state traffic is as follows:

CE1->PE1->**PW2**->PE4->CE2
CE1->PE2->ICL[Split Horizon PW or ICL]->PE1->**PW2**->PE4->CE2

Client Link down protection

When the client link (CL1) is down, the device does not change the MCT status for this VLL. Traffic from the client will be received on CL2 to PE2 and forwarded using EVPN PW from PE2 to PE1. The traffic flow from the client is as follows:

CE1 -> PE2 -> [Split Horizon PW or ICL] -> PE1 -> **PW2** -> PE4 -> CE2

Active MCT Node protection

VLL MCT provides protection when one PE node has a failure including a software or hardware failure, or a power down. In the case when the active MCT node (PE1) is down, the standby MCT node acts as active and uses corresponding PWs for the traffic flow from the client. The traffic flow from the client is as follows:

CE1 -> PE2 -> **PW4** -> PE4 -> CE2

NOTE

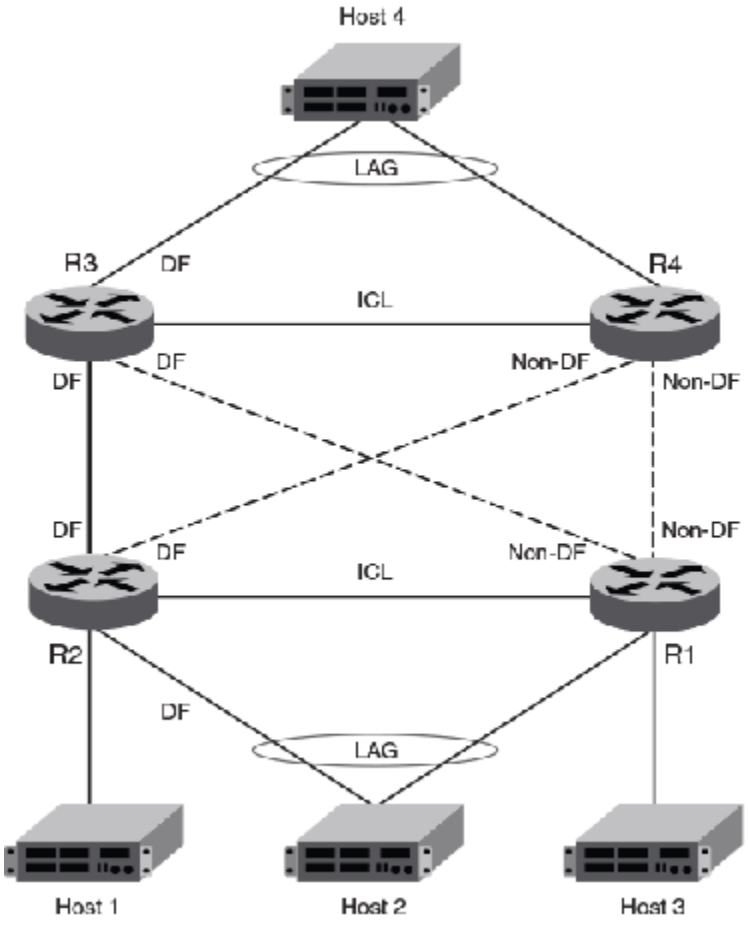
For active-active PW link protection when the PW redundancy status changes, the device relies on the MPLS configuration. There should not be a case where PW2 is down and PW4 is up. The MPLS configuration ensure that both PW2 and PW4 are UP or DOWN. When PW2 is not active-active due to a role change on PE4, PW1 will become active-active.

VPLS-MCT data plane

The main case topology for VPLS MCT is provided in the following figure. Two MCT clusters face each other and four PWs connect the clusters.

NOTE

This topology is for only one BD. Since the DF state is selected per BD, another BD can use a different PE as the active node.

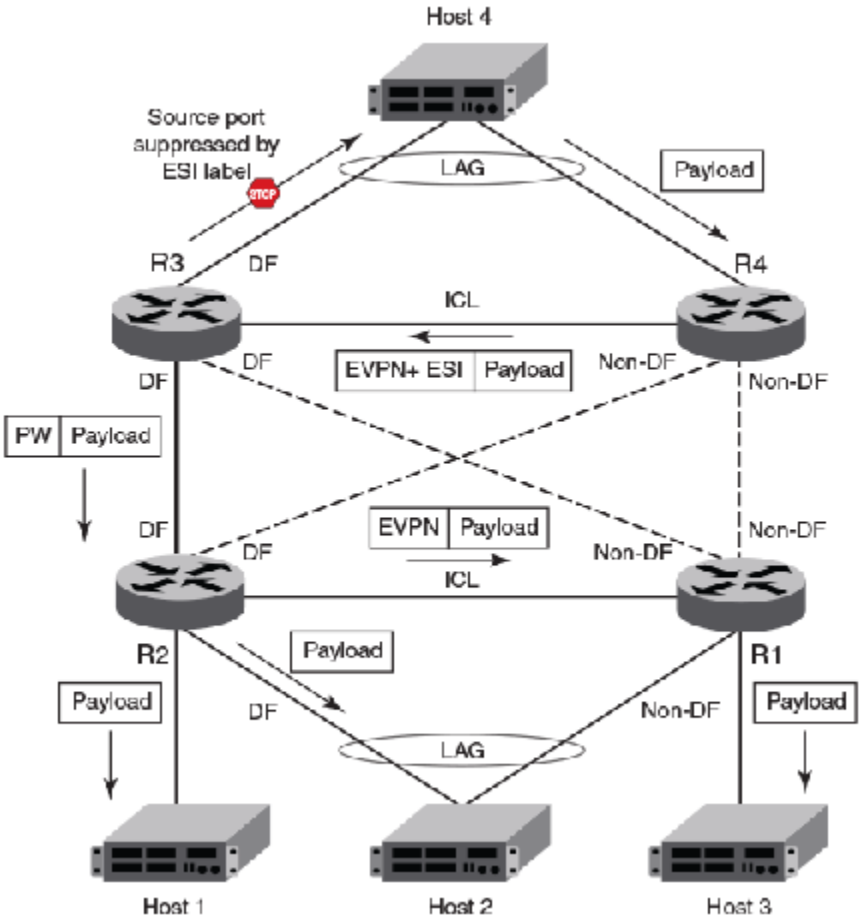


When VPLS MCT is activated, only one PW is operationally active between the MCT clusters, as represented by the solid line. The standby PWs are represented by the dotted lines.

The ICL link between the MCT nodes is a BGP EVPN connection. It is not a spoke PW.

VPLS MCT BUM traffic

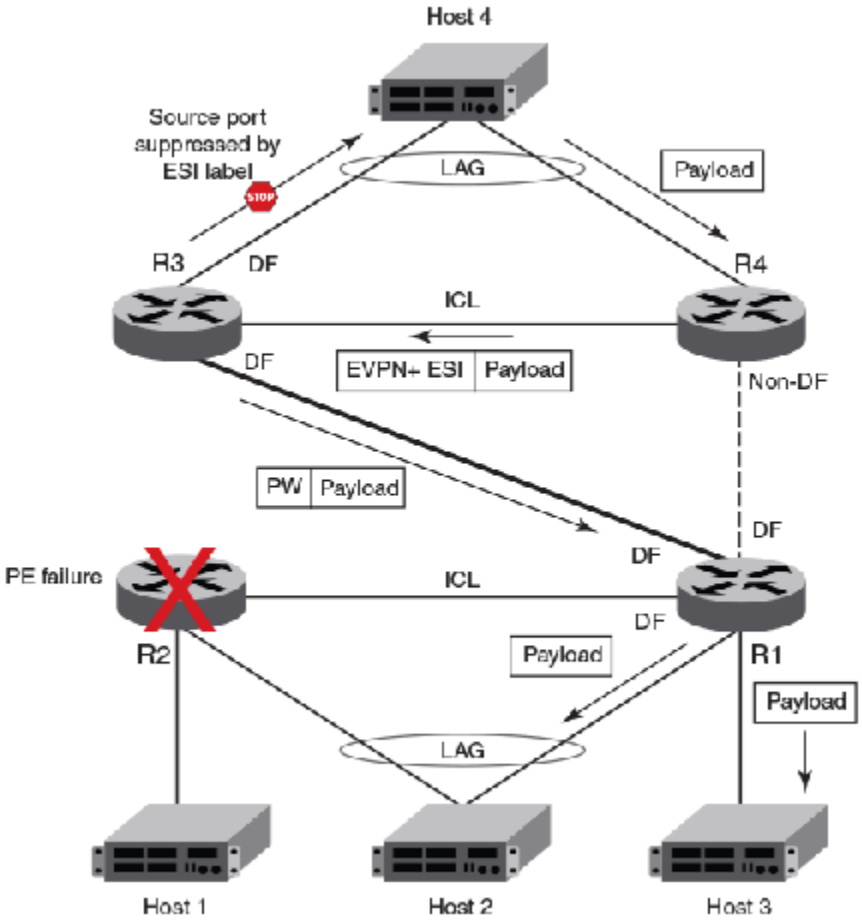
The following figure illustrates how a BUM packet that starts from host 4 travels through the VPLS-MCT network to reach hosts 1, 2, and 3.



VPLS-MCT PE node protection

VPLS MCT provides protection when one PE node has a failure, including a software or hardware failure, or a power-down event. When the active PE fails in MCT, the standby PE becomes the active PE and all PWs on this node transits into MCT active state.

The following figure shows the BUM packet flow after an MCT PE switch-over event.



NOTE

VPLS MCT does not support PW link protection.

VPLS MAC management

In VPLS-MCT topologies, MAC addresses can be learned either on local PW or from the remote MCT node. The same MAC can be learned from either locally or remotely but not from both MCT nodes.

For MCT remote MAC addresses, aging is disabled, and they can only be deleted when delete notifications are received from the remote MCT node that advertised it before for learning.

The following terminologies are associated with MCT MAC entries.

- Dynamic—MAC addresses learned locally as EVPN on CEP ports
- EVPN—MAC addresses learned on remote CEP ports
- Cluster Client Local (CCL)—MAC addresses learned locally on a client interface
- Cluster Client Remote (CCR)—MAC addresses learned on a remote client interface

VPLS MAC addresses that are learned locally are classified as Dynamic, and the corresponding VPLS MAC addresses that are learned remotely are classified as EVPN.

Static MAC handling

Static MAC configuration over the local VPLS endpoints is supported. Static MAC pointing to the PW that is established with the remote VPLS PE is not supported.

MAC learning

MAC addresses learned from the PW on the active PE triggers EVPN MAC synchronization message that are sent to the peer. The PW ESI is used in this MAC route. VPLS CR MAC addresses point to the active MCT node since no local forwarding path on the standby PE traffic is expected to be switched by the active MCT node.

MAC aging

When the VPLS MAC ages on the active node, the MAC address is locally flushed and the EVPN MAC withdrawal route is sent to remote MCT node to flush the MAC.

VPLS MAC movement

A MAC address is considered to be moved when the same MAC address is received on a different interface with same VLAN. In MCT, a MAC movement is allowed on both local and remote nodes.

The following table describes the allowed VPLS MAC movements in MCT.

MAC movement scenario	Behavior
Local dynamic MAC move from PW A to PW B on MCT1.	On local node MCT1, the MAC address is updated to point to the new PW interface. There is no MAC route update required to the remote MCT node. As on the remote node, the MAC always point towards the MCT peer for all VPLS addresses.
Local dynamic MAC move from PW to the Layer 2 CCEP1 client interface on MCT1.	On local node MCT1, the MAC address is updated to point to the new client interface CCEP1. A MAC update route is sent with the new ESI of client 1. The remote node updates the MAC address to point to the CCEP of client 1.
For a MAC learned on a PW locally (MCT1). Dynamic MAC move to CCEP1 on MCT2.	On the MCT2 node for the EVPN MAC learned from MCT1, it is considered as a MAC move when the same mac is learned on a CCEP1 port. The MAC is updated as CCL on MCT2 and now points to the local CCEP1 port on MCT2 instead of pointing to the MCT1 (PW) node. MCT2 sends a CCL MAC update to MCT1. MCT1 updates the MAC as CCR and points to the CCEP1 port.
For a MAC CCL learned on a CCEP1 port locally (MCT1). Dynamic MAC move to the PW on the remote MCT2 node.	On the MCT2 node for the CCR MAC learned from MCT1 for client 1, it is considered as a MAC move when the same MAC is learned on the PW. The MAC is updated as the Dynamic on MCT2 and now points to the PW on MCT2 instead of pointing to CCEP1. From MCT2, it sends a Dynamic MAC update to MCT1. MCT1 updates the MAC to point to MCT2.
Local dynamic MAC move from PW (MCT1) to CEP1 client interface on MCT1.	On local node MCT1, the MAC address is updated to point to the new interface CEP1. A MAC advertise route is sent with ESI 0 to the remote MCT node. The remote node MCT2 updates the MAC address to point to the MCT1 node.
For a MAC learned as EVPN on a PW locally (MCT1). Dynamic MAC move to CEP on MCT2.	On the MCT2 node for the EVPN MAC learned from MCT1, it is considered as a MAC move when the same mac is learned on the CEP port. The MAC is updated as Dynamic on MCT2 and points to the local CEP1 port. From MCT2, it sends a Dynamic MAC updated to MCT1. MCT1 updates the MAC as EVPN and points to the MCT2 node.

MAC movement scenario	Behavior
For a MAC learned as EVPN on a CEP1 port locally (MCT1). Dynamic MAC move to PW on remote MCT2 node.	On MCT2 node for the EVPN MAC learned from MCT1 for CEP, it will be considered as a MAC move when the same mac is learned on the PW. The MAC is updated as Dynamic on MCT2 and now points to the PW on MCT2. MCT2 sends a Dynamic MAC updated to MCT1 with the ESI of the PW client, MCT1 now should updated the MAC point to the MCT2.

MAC address deletion

The following rules are defined for MAC address deletion. Every MAC deletion triggers the MAC resolution algorithm and reprograms the MAC entry if required.

- If a PW is down, MAC addresses flushed locally and individual MAC deletion messages are sent to the MCT Peer. This is similar to the Layer 2 CEP port-down handling.
- If PW client is undeploy on MCT 1, only one MAC withdraw message is send to MCT 2.

  All MAC addresses tagged to the PW client are flushed.
- If MCT 2 detects that MCT 1 is down or if the EVPN session is down, all VPLS MAC addresses that are learned from MCT 1 are flushed.

Configuration considerations and limitations for VPLS and VLL MCT

- Hitless ISSU is not supported. Before starting ISSU, issue the **client-interface shutdown** command on the PE where the ISSU is planned to gracefully move the traffic to the MCT peer. Similarly, use the **force-standby** or **no deploy** command for the PW CCEP before starting ISSU.
- Routing over EVPN is not supported.
- Statistics are not supported.
- For VPLS MCT, consider the following:
  - Configuring a cluster peer as a BD peer impacts data traffic.
  - You can use the **client-interfaces-shutdown** command to shutdown traffic on one node before a software upgrade. After you issue this command, all PWs are put into standby mode.
  - Client-interface shutdown brings down all CCEP interfaces and the BGP session. Other nodes attempt client-isolation logic after the BGP session is down, and you may see the Strict behavior.
  - Logical-interface shutdown brings down the admin state of the CCEP LIF interface if the parent port is an MCT client interface and does not trigger a DF re-election.
  - Protection for a PW link failure is not supported. Active forwarding paths does not occur between the nodes.
- For VLL MCT, consider the following:
  - Cross-connect is used instead of VSI.
  - MAC learning is not required.
  - BUM traffic handling is not required.
  - The MCT role does not depends upon endpoint as well as the PW redundancy status.

Configuring MCT for VPLS or VLL

Configuration of VPLS or VLL for MCT requires the adding of member bridge domains to the MCT cluster and a PW client.

- Before configuring VPLS MCT, configure a point-to-multipoint (p2mp) bridge domain.
- Before configuring VLL MCT, configure a point-to-point (p2p) bridge domain.

For information on configuring VLL or VPLS bridge domains, refer to the "VPLS and VLL Layer 2 VPN services" chapter.

Create an EVPN instance to add the bridge domains and VLANs associated with the MCT cluster. For information on creating the BGP peer, refer to Configuring the BGP EVPN peer on page 124.

For information on configuring the MCT cluster and client, refer to Configuring MCT on page 125. Their full configuration is provided in the example after the steps.

Perform the following steps to configure MCT for VPLS or VLL.

1. In privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Access the cluster on the device.

   ```
   device (config)# cluster MCT1 1
   ```

3. Create the PW client for the cluster and access PW cluster client configuration mode.

   ```
   device(config-cluster-1)# client-pw
   ```

   Only one instance of the PW client represents all VPLS or VLL PWs over all bridge domains.

4. Set the 9-octet Ethernet Segment ID (ESI) value which is used to uniquely identify the PW client.

   ```
   device(config-cluster-client-pw)# esi 01:02:03:04:05:06:07:08:0a
   ```

   You must configure the same value on both MCT nodes.

   The ESI cannot be added under multiple client entries.

5. Deploy the PW client.

   ```
   device(config-cluster-client-pw)# deploy
   ```

6. After configuring the local MCT cluster and PW client, configure the remote MCT cluster and PW client.

The following example is the steps in the previous configuration with the additional configuration of the MCT cluster and client.

```
device# configure terminal
device (config)# cluster MCT1 1
device(config-cluster-1)# peer 10.1.1.1
device(config-cluster-1)# peer-interface Ve 10
device(config-cluster-1)# deploy
device(config-cluster-1)# client MCT1-client 200
device(config-cluster-client-200)# client-interface port-channel 3
device(config-cluster-client-200)# esi 00.a1.b2.c3.d4.e5.f6.89.00
device(config-cluster-client-200)# deploy
device(config-cluster-1)# client-pw
device(config-cluster-client-pw)# esi 01:02:03:04:05:06:07:08:0a
device(config-cluster-client-pw)# deploy
```

Displaying information related to VPLS and VLL MCT

The following examples display PW client and bridge domain information for VPLS and VLL MCT.

Displaying PW client information on an MCT cluster

The following example displays the configuration and state information of the PW client on the MCT cluster.

```
device# show cluster 1
Cluster c1 1
================
```

```
Cluster State: Deployed
Client Isolation Mode: Loose
Configured Member Vlan Range: 100-101
Active Member Vlan Range: 100-101
Configured Member BD Range: 1000-1001
Active Member BD Range: 1000-1001
No. of Peers: 1
No. of Clients: 2


 Peer Info:
 ==========
 Peer IP: 10.38.38.38, State: Up
  Peer Interface: Ethernet 3/1
 ICL Tunnel Type: MPLS, State: Up

 Client Info:
 ============
 Name           Id        ESI                    Interface                Local/Remote State
 ----           ----      -----------            ---------                ------------------
 c3             3         a:b:1:2:3:0:0:0:0:0    Port-channel 100         Up / Up
 Client-PW      34816     a:b:c:d:0:0:0:0:0:0    PW                       Up / Up
```

The following example displays only PW client and its bridge-domain information on the MCT cluster.

```
device# show cluster 1 client-pw
Client Info:
============
Client: Client-pw, client-id: 34816, Deployed, State: Up
Interface: PW
Bridge-domains: 8100-8101
Elected DF for Bridge-domains:
8100
```

The following example displays the multicast and unicast labels, and forwarding state for the cluster member bridge domain.

```
device# show cluster member bridge-domain
 BD-ID     Mcast-label(Lo/Re)    Unicast-label(Lo/Re)    Forwarding state
-------    ------------------    --------------------    ----------------
  1000       822248/ -1                 805864/ 0                 Down
  1001       822249/ -1                 805865/ 0                 Down
```

Displaying the MCT state on a bridge domain

In the **show bridge-domain** output, the MCT Enabled field displays whether the bridge domain is configured under a cluster configuration.

```
device# show bridge-domain 501
Bridge-domain Type: MP , VC-ID: 501, MCT Enabled: TRUE
Number of configured end-points: 3 , Number of Active end-points: 3
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
PW-profile: default, mac-limit: 0
VLAN: 501, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth1/6.501
Un-tagged Ports:
Total VPLS peers: 2 (2 Operational):
VC id: 501, Peer address: 10.9.9.9 , State: Operational , uptime: 1
hr 40 min 51 sec
Load-balance: True , Cos Enabled: False,
Tunnel cnt: 4
rsvp p101 (cos_enable:False cos_value:0)
rsvp p102 (cos_enable:False cos_value:0)
rsvp p103 (cos_enable:False cos_value:0)
rsvp p104 (cos_enable:False cos_value:0)
Assigned LSPs count:4 Assigned LSPs:p101 p102 p103 p104
Local VC lbl: 988042, Remote VC lbl: 985332,
Local VC MTU: 1500, Remote VC MTU: 1500,
Local VC-Type: 5, Remote VC-Type: 5
VC id: 501, Peer address: 56.56.56.56 , State: Operational , uptime: 1
```

```
hr 40 min 13 sec
Load-balance: True , Cos Enabled: False,
Tunnel cnt: 1
rsvp q101 (cos_enable:False cos_value:0)
Assigned LSPs count:0 Assigned LSPs:
Local VC lbl: 988043, Remote VC lbl: 986039,
Local VC MTU: 1500, Remote VC MTU: 1500,
Local VC-Type: 5, Remote VC-Type: 5
```

Displaying MAC address information for a VPLS bridge domain on an MCT cluster

The following example displays the MAC address table for bridge domain of an MCT cluster.

```
device# show mac-address-table bridge-domain
Vlan/BDId    Mac-address       Type           State      Ports/LIF/PeerIp
100(B)       001b.ed0b.ae00    Dynamic        Active      2.2.2.2
100(B)       0230.0774.0aac    Dynamic-CCL     Active      eth 1/2
100(B)       0230.0774.0aac    CCR             Active      po100
200(B)       003d.ed0b.ae00    Dynamic-CCL     Active      3.3.3.3
200(B)       0230.0774.0aac    Dynamic-CCL     Active      3.3.3.3
```

The following example displays all the MAC addresses learned from other VPLS PE nodes over MCT bridge domains.

```
device# show mac-address-table bridge-domain
Vlan/BDId    Mac-address       Type           State      Ports/LIF/PeerIp
100(B)       001b.ed0b.ae00    Dynamic-CCL     Active      2.2.2.2
200(B)       003d.ed0b.ae00    Dynamic-CCL     Active      3.3.3.3
200(B)       0230.0774.0aac    Dynamic-CCL     Active      3.3.3.3
```

Enabling Layer3 routing for an MCT VLAN

Layer 3 routing is supported for IPv4 and IPv6 BGP, OSPF, and IS-IS routing protocols on an MCT VLAN.

The enabling of the Layer 3 protocols on the MCT VLAN are the same as enabling them on a VE interface. You must first create the VE interface for an MCT VLAN.

> **NOTE**
> For VE over an MCT VLAN interface, you cannot enable MPLS on it.

The following configuration example enables OSPFv2 and OSPFv3 protocols on VE 200 for the MCT member VLAN 2.

```
router ospf
  area 0

ipv6 router ospf
  area 0

vlan 2
 router-interface Ve 200

interface Ve 200
 ipv6 address 2001::1/64
 ip address 10.2.2.1/24

 ip ospf area 0
 ipv6 ospf area 0
 !
 no shutdown
!
```

Using MCT with VRRP and VRRP-E

Standard VRRP and VRRP-E configuration commands, VMAC generation login, and scaling numbers apply when used with MCT. VRRP advertisement packets are exchanged through the MCT ICL.

The MCT device that acts as the Virtual Routing Redundancy Protocol (VRRP) and VRRP Extended (VRRP-E) backup router performs as a Layer 2 switch to pass the packets to the VRRP or VRRP-E master router for forwarding. Through MAC synchronization, the VRRP or VRRP-E backup router learns the virtual MAC (VMAC) on the Inter-Chassis Link (ICL) represented by the MPLS cloud in the diagram. The data traffic and control traffic both pass through the ICL MPLS cloud link from the backup router. If VRRP-E short path forwarding is enabled, the backup router can forward the packets directly, instead of sending them to the master.

NOTE
Short path forwarding is only supported on VRRP-E.

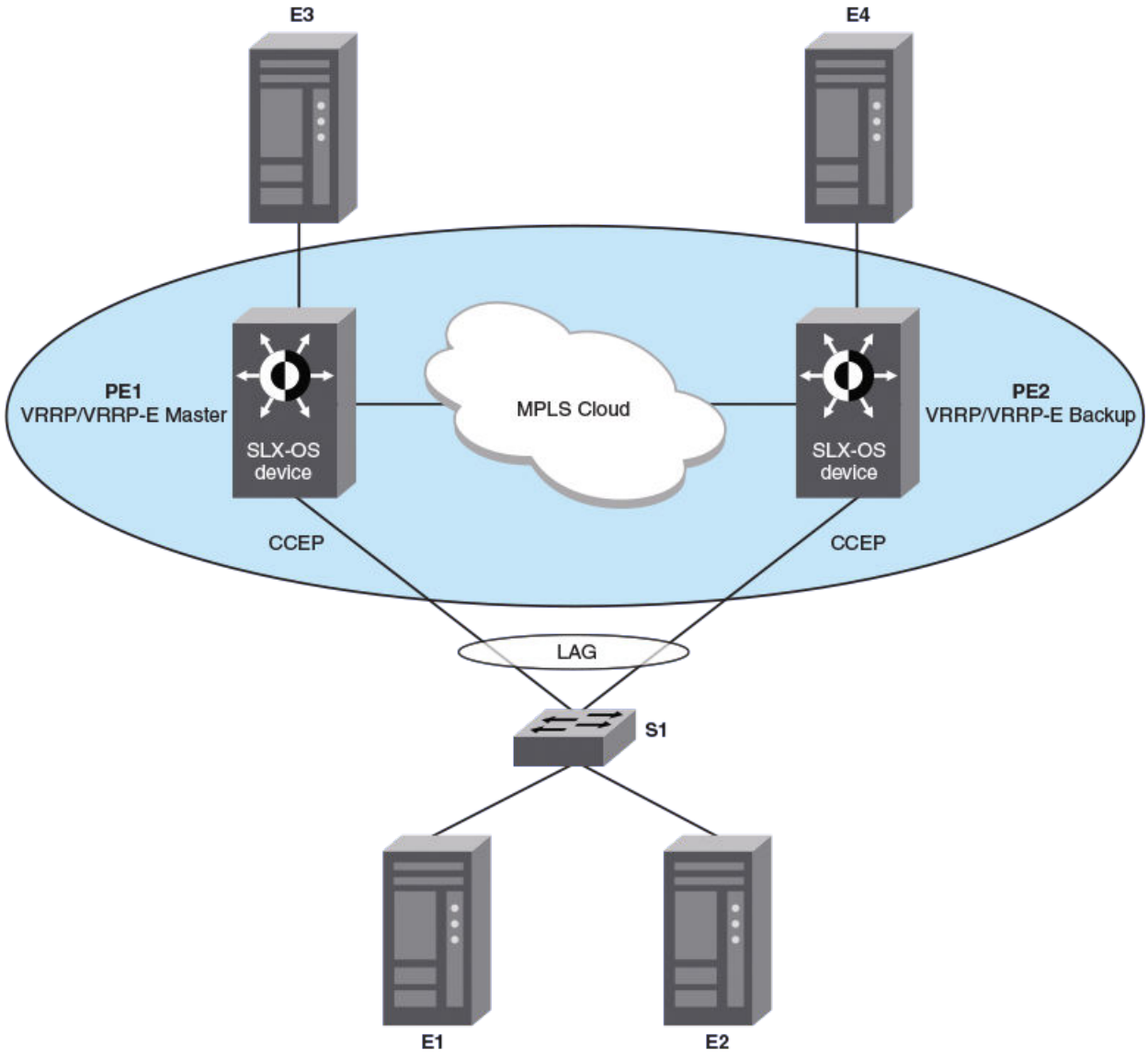
In the diagram below, when an ARP request from the S1 switch device is sent through the direct link to the VRRP or VRRP-E backup router (PE2), a broadcast packet is sent to the VRRP/E master router (PE1) for processing through the ICL (MPLS cloud). When the ARP request is received by the PE1 device, PE1 sends a reply through the direct link to S1. If the ARP reply was received before the MAC address for the MCT on S1 is learned, the reply packet may be flooded to both the Customer Client Edge Port (CCEP) ports and ICL ports.

Using VRRP or VRRP-E, data traffic received from a client device on a backup router is Layer 2 switched to the master device. If VRRP-E short path forwarding is enabled, traffic received on the backup device may be forwarded by the backup if the route to the destination device is shorted than through the master device.

MCT short path forwarding configuration using VRRP-E example

In this example configuration, we are assuming that MCT is using the VRRP-E short path forwarding. When short path forwarding is enabled, packets from either the E1 or E2 devices with a destination of the E4 device can be routed through the PE 2 device which is a VRRP-E backup device. Short path forwarding is designed for load-balancing and allows packets to use the shortest path, and in this case, PE2 is directly connected to E4 so the packets will travel through PE2.

FIGURE 23 MCT short path forwarding



PE1 configuration

The following example configures the OSPF, BGP, and MPLS protocols with cluster configuration for MCT for the PE1 router in the diagram. A VRRP-E priority value of 110 (higher than the device at PE2) allows the PE1 device to assume the role of VRRP-E master.

```
ip router-id 10.19.19.19
 router ospf
   area 0

 interface Loopback 200
  no shutdown
  ip ospf area 0
  ip address 10.19.19.19/32

 router bgp
 local-as 100
```

```
 neighbor 10.32.32.32 remote-as 100
 neighbor 10.32.32.32 update-source loopback 200
 address-family ipv4 unicast
!
 address-family ipv6 unicast
 address-family l2vpn evpn
 neighbor 10.32.32.32 activate
 exit
!
 interface Ethernet 2/3
  ip address 10.1.8.19/24
  ip ospf area 0
  no shutdown
!
router mpls
 mpls-int Ethernet 2/3
 lsp to32
  to 10.32.32.32
  enable
!
vlan 100
!
interface Ethernet 2/5
 switchport
 switchport mode trunk-no-default-native
 switchport trunk allow vlan add 100
 no shutdown
!
 evpn myinstance
  vlan add 100
  rd auto
  route-target both auto ignore-as
!
cluster c1 1
 peer 10.32.32.32
 deploy
  client c1 1
   client-interface Ethernet 2/5
    esi 01:02:03:04:05:06:07:08:09
    deploy
!
vlan 100
 router-interface Ve 100
!
protocol vrrp-extended
interface Ve 100
 ip proxy-arp
 ip address 10.2.3.6/24
 vrrp-extended-group 1
  priority 110
  short-path-forwarding
  virtual-ip 10.2.3.4
 no shutdown
!
interface Ve 100
 ipv6 address fe80::1:2 link-local
 ipv6 address 3313::2/64
 ipv6 vrrp-extended-group 1
 virtual-ip 3313::1
```

PE2 configuration

The following example configures the OSPF, BGP, and MPLS protocols with cluster configuration for MCT for the PE2 router in the diagram. A VRRP-E priority value of 80 (lower than the device at PE1) allows the PE2 device to assume the role of a VRRP-E backup device.

```
ip router-id 10.32.32.32
 router ospf
```

```
  area 0

 interface Loopback 100
  no shutdown
  ip ospf area 0
  ip address 10.32.32.32/32

 router bgp
 local-as 100
 neighbor 10.19.19.19 remote-as 100
 neighbor 10.19.19.19 update-source loopback 100
 address-family ipv4 unicast
!
 address-family ipv6 unicast
!
 address-family l2vpn evpn
 neighbor 10.19.19.19 activate
!
!
interface Ethernet 2/3
 ip address 10.1.8.32/24
 no shutdown
  ip ospf area 0
!
router mpls
 mpls-int Ethernet 2/3
  lsp to19
   to 10.19.19.19
   enable
!
vlan 100
!
interface Ethernet 2/7
 switchport
 switchport mode trunk-no-default-native
 switchport trunk allow vlan add 100
 no shutdown
!
 evpn myinstance
  vlan add 100
  rd auto
  route-target both auto ignore-as
!
cluster c1 1
 peer 10.19.19.19
 deploy
  client c1 1
    esi 01:02:03:04:05:06:07:08:09
    client-interface Ethernet 2/7
    deploy
!
vlan 100
 router-interface Ve 100
!
protocol vrrp-extended
interface Ve 100
 ip proxy-arp
 ip address 10.2.3.5/24
 vrrp-extended-group 1
  priority 80
  short-path-forwarding
  virtual-ip 10.2.3.4
 no shutdown
!
interface Ve 100
 ipv6 address fe80::1:1 link-local
 ipv6 address 3313::3/64
 ipv6 vrrp-extended-group 1
 virtual-ip 3313::1
```

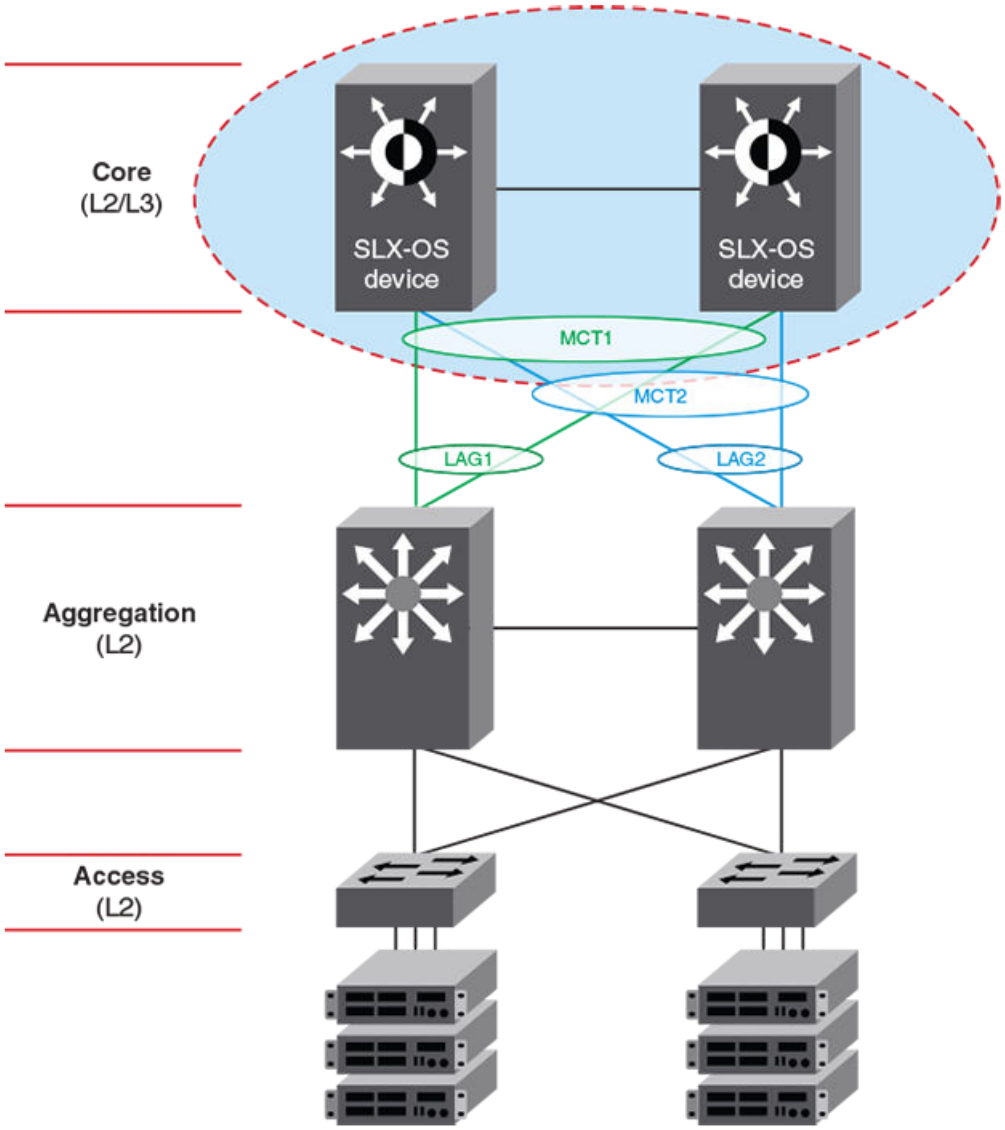
MCT use cases

An L2 MCT solution can be deployed at the access, aggregation, and the core of the data center. However, SLX-OS device is targeted for the data center core.

L2 MCT in the data center core

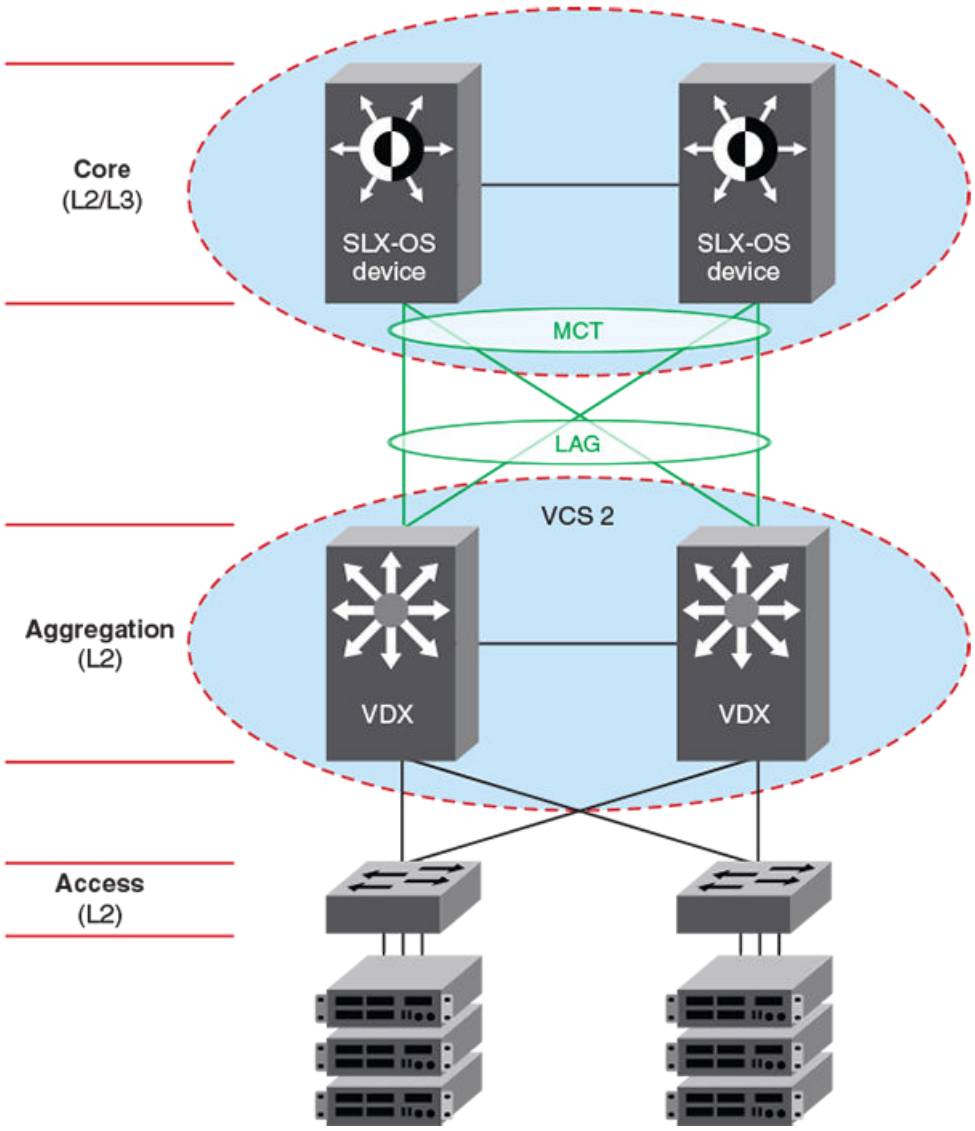
The following diagram shows a typical 3-tier data center where access and aggregation layers are running Layer 2 and the core is running Layer 2 and Layer 3. The access and aggregation can be standalone Extreme switches or any other third party switches.

FIGURE 24 Typical 3-tier data center



Another variation of this use case is when the aggregation layer is a virtual cluster of switches which is transparent to SLX-OS devices in the core layer.

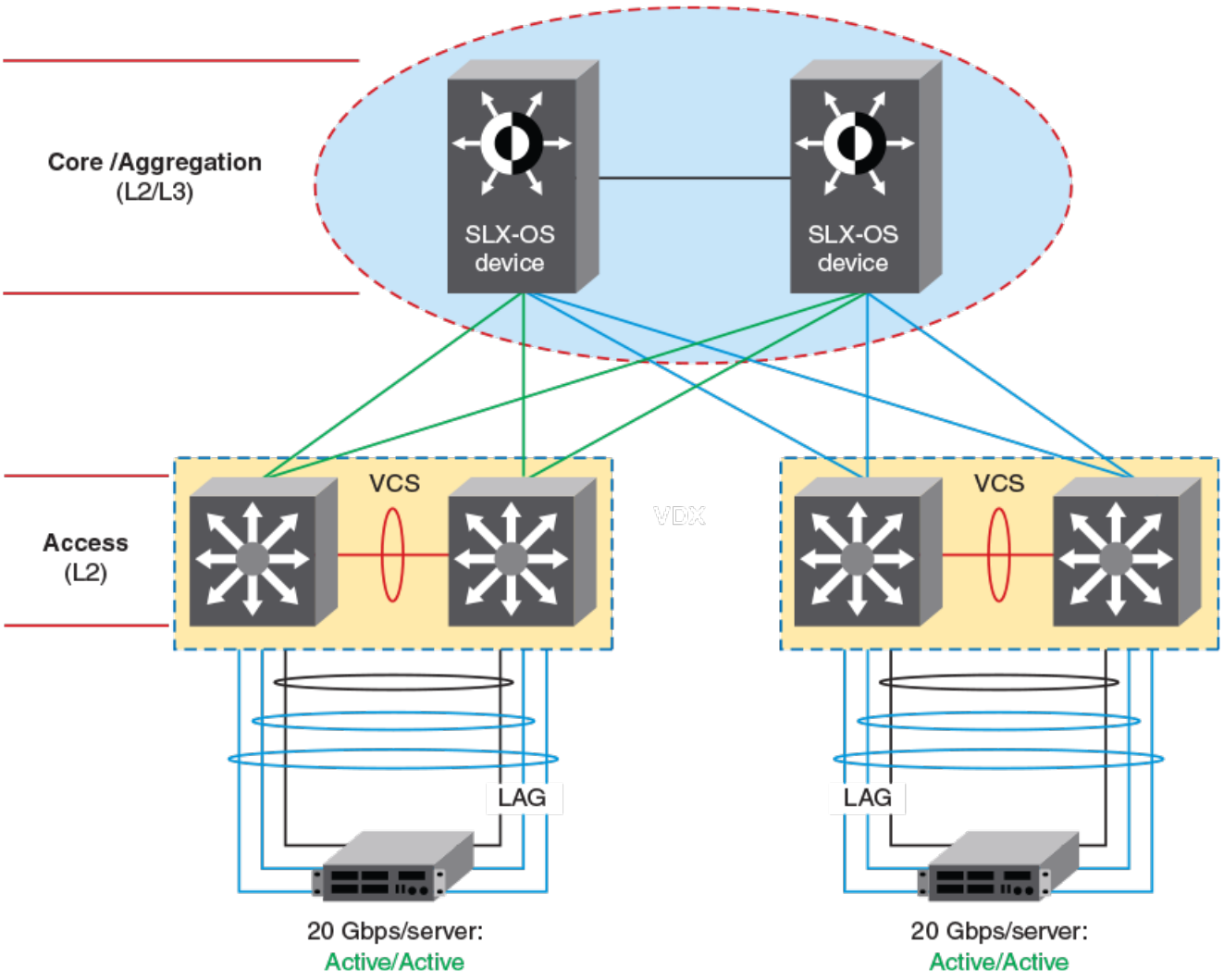
FIGURE 25 L2 MCT in the data center core connecting to a VCS



L2 MCT in a data center with a collapsed core and aggregation

The following diagram describes a scenario where VCS fabric of VDX 8870 switches is deployed at the access layer. With the availability of 10G and 40G interface, access switches can connect directly to the core without the need to have a separate aggregation layer.

FIGURE 26 L2 MCT with collapsed core and aggregation



Logical Interfaces

Logical interfaces overview

A logical interface (LIF) serves the purpose of abstracting a forwarding interface in a very generic way, making it possible to capture the underlying physical characteristics of a forwarding interface.

This feature facilitates the support of future forwarding technologies without the need to modify code design in various software components.

A forwarding interface is also known as "main interface." It can be a physical port, a port-channel (Link Aggregation Group, or LAG), a pseudowire (PW), a tunnel, and so on. A logical interface can also be thought of as a subinterface configuration on top of a main interface.

NOTE
Currently the only LIFs that require explicit user configuration are attachment circuit (AC) LIFs.

LIFs and bridge domains

A Layer 2 application for LIFs is for bridge domains (BDs). A BD is an infrastructure that supports the implementation of different switching technologies; it is essentially a generic broadcast/forwarding domain that is not tied to a specific transport technology. Bridge domains support a wide range of service endpoints, including regular Layer 2 endpoints and Layer 2 endpoints over Layer 3 technologies. Logical interfaces representing BD endpoints must be created before they can be bound to a BD. For more information and configuration details, refer to the Bridge Domains on page 153 chapter in this guide.

Configuration considerations

The following are some common rules to consider in configuring logical interfaces:

- By default, when the LIF is created it is configured as "no shutdown."
- By default, when the LIF is created, it is "tagged" unless it is explicitly configured with the "untagged" option.
- Allowed LIF service instance ID ranges are from 1 through 12288.
- An LIF service instance ID has no correlation to the VLAN ID of the LIF.
- Each physical/LAG-based LIF must have an associated VLAN configured or else it will not be usable when the user attempts to add it to a service (such as VPLS, Layer 2). Such a configuration request to add the LIF to a service will be rejected.
- Once the LIF is associated with a Layer 2 service, its VLAN value cannot be changed or deleted unless it is first removed from the associated service. In case the LIF is not yet associated to a service, the user is free to remove the VLAN configuration or change the VLAN assignment.
- The **no** option to the **logical-interface** command can be applied at any time.
- The "untagged" configuration is allowed for only one LIF under the same physical port or LAG. If one LIF is already configured as untagged, all subsequent attempts on the same physical port or LAG are rejected.
- Once the "untagged" option is selected, it will only have one VLAN as the next classification option. There is no dual-tag support for the untagged case.

- In order to configure an untagged LIF, the main interface must be configured as "switchport mode trunk-no-default-native". If it is configured set to regular trunk mode, the native VLAN is already associated with a regular Layer 2 VLAN LIF and no explicit untagged LIF can be configured on that interface.
- Once the LIF is associated with a service (Layer 2) such as a bridge domain, its "untagged/tagged" configuration cannot be changed. The service instance or its current VLAN classification must be deleted by the user first and then added back with the proper "untagged/tagged" option.
- VLANs 4091 through 4095 are reserved VLANs and these should not be used as the VLAN ID for either the inner or outer VLAN of the LIF.
- The VLAN specified under the LIF ensures that such a VLAN is not already configured under the **switchport** command for a regular Layer 2 allowed VLAN.

If the interface is already configured as "switchport access," then it is not allowed to be configured with LIF. The reverse condition is also not allowed: the interface cannot be changed to mode access if a LIF is still configured under the main interface.

Configuring a logical interface on a physical port or port-channel (LAG)

This task configures a logical interface on a physical (Ethernet) port and a port-channel (Link Aggregation Group, or LAG) interface.

Refer to the Usage Guidelines for the **logical-interface** command for complete details.

1. Do the following to configure a logical interface on an Ethernet port.
   a) Enter global configuration mode.

      ```
      device# configure terminal
      ```

   b) Specify an Ethernet interface.

      ```
      device(config)# interface ethernet 2/6
      ```

   c) Enter the **switchport** command to configure the parent interface as switchport.

      ```
      device(conf-if-eth-2/6)# switchport
      ```

   d) Enter the **switchport mode trunk-no-default-native** command to enable an explicit untagged LIF to be configured.

      ```
      device(conf-if-eth-2/6)# switchport mode trunk-no-default-native
      ```

   e) Enable the interface.

      ```
      device(conf-if-eth-2/6)# no shutdown
      ```

   f) Enter the **logical-interface** command, specify a service instance, and enter LIF configuration mode.

      ```
      device(conf-if-eth-2/6)# logical-interface ethernet 2/6.120
      ```

   g) (Optional) Enter the **name** command to facilitate the management of the LIF.

      ```
      device(conf-if-eth-lif-2/6.120)# name myLIF120
      ```

   h) Enter the **vlan** command with the **inner-vlan** option to specify an interface and create dual-tag VLANs.

      ```
      device(conf-if-eth-lif-2/6.120)# vlan 120 inner-vlan 200
      ```

   i) Alternatively, enter the **untagged vlan** command to specify that the LIF is to receive untagged packets.

      ```
      device(conf-if-eth-lif-2/6.120)# untagged vlan 120
      ```

      See the Usage Guidelines for the **vlan (LIF)** command.

   j) (Optional) By default, the administrative state of the LIF is "no shutdown." To remove the port from participating in any data traffic without having to shut down the physical interface, enter the **no** form of the **shutdown (LIF)** command.

      ```
      device(conf-if-eth-lif-2/6.120)# no shutdown
      ```

   k) (Optional) For convenience, you can also enter up to two options in a single command line, as in the following examples.

      ```
      device(conf-if-eth-2/6)# logical-interface ethernet 2/6.120 name myLIF120
      device(conf-if-eth-2/6)# logical-interface ethernet 2/6.120 vlan 120
      ```

2. To configure a port-channel, configure the basic LIF parameters and options as in Step 1.
   a) Specify a port-channel, set its mode to "trunk-no-default-native," and specify a logical interface service instance.

```
device(config)# interface port-channel 10
device(config-port-channel-10)# switchport mode trunk-no-default-native
device(config-port-channel-10)# logical-interface port-channel 10.3
device(config-if-po-lif-10.3)#
```

   b) Repeat additional substeps in Step 1 as appropriate.

Bridge Domains

Bridge domain overview

Bridge domain is an infrastructure that supports the implementation of different switching technologies.

A bridge domain is a generic broadcast domain that is not tied to a specific transport technology. Bridge domains support a wide range of service endpoints including regular L2 endpoints and L2 endpoints over L3 technologies.

Bridge domains switch packets between a range of different endpoint types; for example, attachment circuit (AC) endpoints, Virtual Private LAN Service (VPLS) endpoints, Virtual Leased Line (VLL) endpoints, and tunnel endpoints.

The following are examples of bridge-domain capable services:

- VPLS—with multiple AC endpoints and pseudowire (PW) logical interfaces (LIFs)
- Local VPLS—with multiple AC endpoints
- VLL—with one AC endpoint and one PW endpoint
- VLL—with two AC endpoints

A bridge domain that is created for a VPLS application is also referred to as a VPLS instance.

Bridge domain statistics

Devices gather statistics for all the logical interfaces and peers in bridge domains.

Statistics must be manually enabled for a specific bridge domain, since statistics for bridge domains are not enabled by default.

Use the **statistics** command in bridge domain configuration mode to enable statistics on a bridge domain.

NOTE

- The statistics reported are not real-time statistics since they depend upon the load on the system.
- Enabling statistics on a bridge domain has a heavy impact on data traffic.
- For optimum utilization of the statistics resources in the hardware, statistics on a bridge domain are not enabled by default.

Configuring a bridge domain

Bridge domains are configured independently of the different switching technologies that they support.

Before configuring a bridge domain, configure any logical interface that is to be bound to the bridge domain. Logical interfaces that represent bridge-domain endpoints must be created before they are bound to a bridge domain. For further information on configuration of logical interfaces, refer to *Logical Interfaces*.

There is an example at the end of this task that shows all the configuration steps in order.

Perform the following task to configure a bridge domain.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Create a bridge domain.

```
device(config)# bridge-domain 5 p2p
```

   By default, the bridge-domain service type is multipoint (**p2mp**). In this example, bridge domain 5 is configured as a point-to-point service (**p2p**).

3. **NOTE**
   Logical interfaces representing bridge-domain endpoints must be created before they can be bound to a bridge domain. For further information, refer to *Logical Interfaces*.

   Bind the logical interfaces for attachment circuit endpoints to the bridge domain.

```
device(config-bridge-domain-5)# logical-interface ethernet 1/6.400
```

   In this example, Ethernet logical interface 1/6.400 is bound to bridge domain 5.

4. Repeat Step 4 to bind other logical interfaces for attachment circuit endpoints to the bridge domain.

```
device(config-bridge-domain-5)# logical-interface port-channel 2.200
```

   In this example, port channel logical interface 2.200 is bound to bridge domain 5.

5. (Optional) Enable local switching for bridge domain 5.

```
device(config-bridge-domain-5)# local-switching
```

   By default, local switching is enabled.

6. (Optional) Enable dropping L2 bridge protocol data units (BPDUs) for bridge domain 5.

```
device(config-bridge-domain-5)# bpdu-drop-enable
```

The following example creates bridge domain 5. It binds ethernet and port-channel logical interfaces to the bridge domain. It configures local switching, and enables dropping of L2 BPDUs.

```
device# configure terminal
device(config)# bridge-domain 5
device(config-bridge-domain-5)# logical-interface ethernet 1/6.400
device(config-bridge-domain-5)# logical-interface port-channel 2.200
device(config-bridge-domain-5)# local-switching
device(config-bridge-domain-5)# bpdu-drop-enable
```

Displaying bridge-domain configuration information

Various show commands can be used to display bridge-domain configuration information.

- Enter the **show bridge-domain** command to display information about all configured bridge domains.

```
device# show bridge-domain

Total Number of bridge-domains: 3
Number of bridge-domains: 3

Bridge-domain 1
-------------------------------
```

```
Bridge-domain Type: mp , VC-ID: 5
Number of configured end-points:  5 , Number of Active end-points: 4
VE if-indx: 1207959555, Local switching: TRUE, bpdu-drop-enable:TRUE
PW-profile: 1, mac-limit: 128000
Number of Mac's learned:90000,     Static-mac count: 10,
VLAN: 100, Tagged ports: 2(2 up), Un-tagged ports: 0 (0 up)
Tagged ports: Eth 0/2/6, eth 0/2/8
Un-tagged ports:

Total PW peers: 2 (2 Operational)
Peer address: 12.12.12.12, State: Operational, Uptime: 2 hr 55 min
      Load-balance: True , Cos enabled:False,
      Assigned LSP;s:
      Tnnl in use: tnl2[RSVP]
      Local VC lbl: 983040, Remote VC lbl: 983040
      Local VC MTU: 1500, Remote VC MTU: 1500,
      Local VC-Type: Ethernet(0x05), Remote VC-Type: Ethernet(0x05)
Peer address: 15.15.15.15, State: Operational, Uptime: 2 hr 55 min
         Load-balance: False , Cos enabled:False,
      Assigned LSP's: lsp1, lsp2
      Tnnl in use: tnl1[MPLS]
      Local VC lbl: 983041, Remote VC lbl: 983043
      Local VC MTU: 1500, Remote VC MTU: 1500 ,
      Local VC-Type: Ethernet(0x05), Remote VC-Type: Ethernet(0x05)

Bridge-domain 2
-------------------------------
Bridge-domain Type: mp , VC-ID: 100
Number of configured end-points:  5 , Number of Active end-points: 4
VE if-indx:  NA, Local switching: FALSE, bpdu-drop-enable:FALSE
PW-profile: profile_1, mac-limit: 262144
Number of Mac's learned:90000,     Static-mac count: 10,
VLAN: 100, Tagged ports: 2(1 up), Un-tagged ports: 0 (0 up)
      Tagged ports: eth 0/2/10, eth 0/1/10
      Un-tagged ports:
VLAN: 150, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
      Tagged ports: eth 0/1/5
      Un-tagged ports:

Bridge-domain 3
-------------------------------
Bridge-domain Type: mp , VC-ID: 200
Number of configured end-points:  5 , Number of Active end-points: 4
VE if-indx: 120793855, Local switching: FALSE, bpdu-drop-enable:FALSE
PW-profile: 2, mac-limit: 262144
Number of Mac's learned:90000,     Static-mac count: 10,
Local switching: TRUE,
VLAN: 500, Tagged ports: 2(2 up), Un-tagged ports: 2 (1 up)
Tagged ports:     eth 0/11/6, eth 0/4/3
Un-tagged ports:

Total VPLS peers: 3 (2 Operational)
Peer address: 5.5.5.5, State: Operational, Uptime: 2 hr 35 min
         Load-balance: False , Cos enabled:False,
      Assigned LSP;s:
      Tnnl in use: tnl2[RSVP]
      Local VC lbl: 983050, Remote VC lbl: 983050
      Local VC MTU: 1500,Remote VC MTU: 1500,
      Local VC-Type: Ethernet(0x05), Remote VC-Type: Ethernet(0x05)
Peer address: 20.20.20.20, State: Operational, Uptime: 0 hr 18 min
         Load-balance: False , Cos enabled:True,
      Assigned LSP's:
      Tnnl in use: NA,
      Local VC lbl: NA, Remote VC lbl: NA
      Local VC MTU: 1500,Remote VC MTU: 1500,
      Local VC-Type: Ethernet(0x05), Remote VC-Type: Ethernet(0x05)
Peer address: 10.10.10.10, State: Not-Operational (Tunnel Not Available),
         Load-balance: True , Cos enabled:False,
      Assigned LSP's: lsp10, lsp15
      Tnnl in use: NA,
      Peer Index:2
      Local VC lbl: NA, Remote VC lbl: NA
```

```
      Local VC MTU: 1500,Remote VC MTU: NA ,
      Local VC-Type: Ethernet(0x05), Remote VC-Type: NA
```

- Enter the **show bridge-domain** command specifying the bridge-domain ID to display information about a specific bridge domain. The following example displays information about bridge domain 501.

```
device# show bridge-domain 501

Bridge-domain 501
------------------------------
Bridge-domain Type: MP , VC-ID: 501
Number of configured end-points:  2 , Number of Active end-points: 2
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
PW-profile: default, mac-limit: 0
VLAN: 501, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth1/6.501
Un-tagged Ports:
Total VPLS peers: 1 (1 Operational):

VC id: 501, Peer address: 10.9.9.9, State: Operational, uptime: 2 sec
      Load-balance: False, Cos Enabled: False,
      Tunnel cnt: 1
      rsvp  p101(cos_enable:False cos_value:0)
      Assigned LSPs count:0 Assigned LSPs:
      Local VC lbl: 989042, Remote VC lbl: 983040,
      Local VC MTU: 1500, Remote VC MTU: 1500,
      Local VC-Type: 5, Remote VC-Type: 5
```

The following example shows information about a bridge domain (501) in which the **load-balance** option is configured for the peer device 10.9.9.9.

```
show bridge-domain 501

Bridge-domain 501
------------------------------
Bridge-domain Type: MP , VC-ID: 501
Number of configured end-points:  2 , Number of Active end-points: 2
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
PW-profile: default, mac-limit: 0
VLAN: 501, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth1/6.501
Un-tagged Ports:
Total VPLS peers: 1 (1 Operational):

VC id: 501, Peer address: 10.9.9.9, State: Operational, uptime: 48 sec
      Load-balance: True , Cos Enabled: False,
      Tunnel cnt: 16
      rsvp  p101(cos_enable:False cos_value:0)
      rsvp  p102(cos_enable:False cos_value:0)
      rsvp  p103(cos_enable:False cos_value:0)
      rsvp  p104(cos_enable:False cos_value:0)
      rsvp  p105(cos_enable:False cos_value:0)
      rsvp  p106(cos_enable:False cos_value:0)
      rsvp  p107(cos_enable:False cos_value:0)
      rsvp  p108(cos_enable:False cos_value:0)
      rsvp  p109(cos_enable:False cos_value:0)
      rsvp  p110(cos_enable:False cos_value:0)
      rsvp  p111(cos_enable:False cos_value:0)
      rsvp  p112(cos_enable:False cos_value:0)
      rsvp  p113(cos_enable:False cos_value:0)
      rsvp  p114(cos_enable:False cos_value:0)
      rsvp  p115(cos_enable:False cos_value:0)
      rsvp  p116(cos_enable:False cos_value:0)
      Assigned LSPs count:0 Assigned LSPs:
      Local VC lbl: 989040, Remote VC lbl: 983040,
      Local VC MTU: 1500, Remote VC MTU: 1500,
      Local VC-Type: 5, Remote VC-Type: 5
```

The following example shows information about bridge domain 501 in which the **load-balance** option and four assigned label-switched paths (p101, p102, p103, and p104) are configured for the peer device 10.9.9.9.

```
device# show bridge-domain 501

Bridge-domain 501
------------------------------
Bridge-domain Type: MP , VC-ID: 501
Number of configured end-points:  2 , Number of Active end-points: 2
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
PW-profile: default, mac-limit: 0
VLAN: 501, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth1/6.501
Un-tagged Ports:
Total VPLS peers: 1 (1 Operational):

VC id: 501, Peer address: 10.9.9.9, State: Operational, uptime: 4 sec
      Load-balance: True , Cos Enabled: False,
      Tunnel cnt: 4
      rsvp  p101(cos_enable:False cos_value:0)
      rsvp  p102(cos_enable:False cos_value:0)
      rsvp  p103(cos_enable:False cos_value:0)
      rsvp  p104(cos_enable:False cos_value:0)
      Assigned LSPs count:4 Assigned LSPs:p101 p102 p103 p104
      Local VC lbl: 989041, Remote VC lbl: 983040,
      Local VC MTU: 1500, Remote VC MTU: 1500,
      Local VC-Type: 5, Remote VC-Type: 5
```

- Enter the **show bridge-domain brief** command to display summary information about all configured bridge domains.

```
device# show bridge-domain brief

Total Number of bridge-domains configured: 10
Number of VPLS bridge-domains: 5
Macs Dynamically learned: 50360, Macs statically configured: 0

BDID(VC-ID)   TYPE     Intf(up)     PWs(up)  macs
501(501)      P2MP     5(3)         2(2)     50000
502(502)      P2MP     1(1)         1(1)     10
503(503)      P2MP     10(6)        3(1)     0
504(504)      P2MP     1(1)         1(1)     350
505(505)      P2MP     1(1)         1(1)     0
506(506)      P2P      1(1)         1(1)     0
507(507)      P2P      1(1)         1(1)     0
508(508)      P2P      1(1)         1(1)     0
509(509)      P2P      1(1)         1(1)     0
510(510)      P2P      1(1)         1(1)     0
```

Enabling statistics on a bridge domain

Statistics may be enabled for a specific bridge domain.

NOTE

By default statistics are disabled on bridge domains. After enablement, statistics should be disabled when no longer needed because the collection of statistical information has a heavy impact on data traffic.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Enter the **bridge-domain** command to create a bridge domain at the global configuration level.

```
device(config)# bridge-domain 3
```

3. Enter the **statistics** command to enable statistics for all the logical interfaces and peers in the bridge domain.

```
device(config-bridge-domain-3)# statistics
```

NOTE

When statistics are no longer needed, use the **no statistics** command to disable statistics on the bridge domain.

The following example shows how to enable statistics on bridge domain 3.

```
device# configure terminal
device(config)# bridge-domain 3
device(config-bridge-domain-3)# statistics
```

The following example shows how to disable statistics on bridge domain 3.

```
device# configure terminal
device(config)# bridge-domain 3
device(config-bridge-domain-3)# no statistics
```

Displaying statistics for logical interfaces in bridge domains

Various commands can be used to display statistical information for bridge domains.

- Enter the **show statistics bridge-domain** command to display statistics for all logical interfaces and peers on all configured bridge domains.

```
device# show statistics bridge-domain

Bridge Domain 1 Statistics
Interface                    RxPkts          RxBytes          TxPkts          TxBytes
eth 1/1.100       821729          821729          95940360        95940360
eth 1/21.200      884484          885855          95969584        95484555
po  1.300           8884            8855              9684            9955

Bridge Domain 20 Statistics
Interface                    RxPkts          RxBytes          TxPkts          TxBytes
eth 1/6.400       821729          821729          95940360        95940360
eth 1/21.100        8884            8855              9684            9955
po  2.40          884484          885855          95969584        95484555
```

- Enter the **show statistics bridge-domain** command specifying a bridge-domain ID to view the statistics for a specific bridge domain. The following example displays statistics for bridge-domain ID 1.

```
device# show statistics bridge-domain 1

Bridge Domain 1 Statistics
Interface                    RxPkts          RxBytes          TxPkts          TxBytes
eth 1/1.100       821729          821729          95940360        95940360
eth 1/21.200      884484          885855          95969584        95484555
po  1.300           8884            8855              9684            9955
```

Clearing statistics on bridge domains

Statistical information can be cleared for all bridge domains or for a specific bridge domain.

- Enter the **clear statistics bridge-domain** command to clear statistics for all logical interfaces and peers on all configured bridge domains.

```
device# clear statistics bridge-domain
```

- Enter the **clear statistics bridge-domain** command specifying the bridge-domain ID to clear the statistics for a specific bridge domain. The following example shows how to clear statistics for bridge domain ID 1.

```
device# clear statistics bridge-domain 1
```

VPLS and VLL Layer 2 VPN services

VPLS overview

Virtual Private LAN Service (VPLS) is a Layer 2 Virtual Private Network (L2 VPN) architecture that provides multipoint Ethernet LAN services.

VPLS provides transparent LAN services across provider edge (PE) devices using Internet Protocol (IP) or Multiprotocol Label Switching (MPLS) as the transport technology.

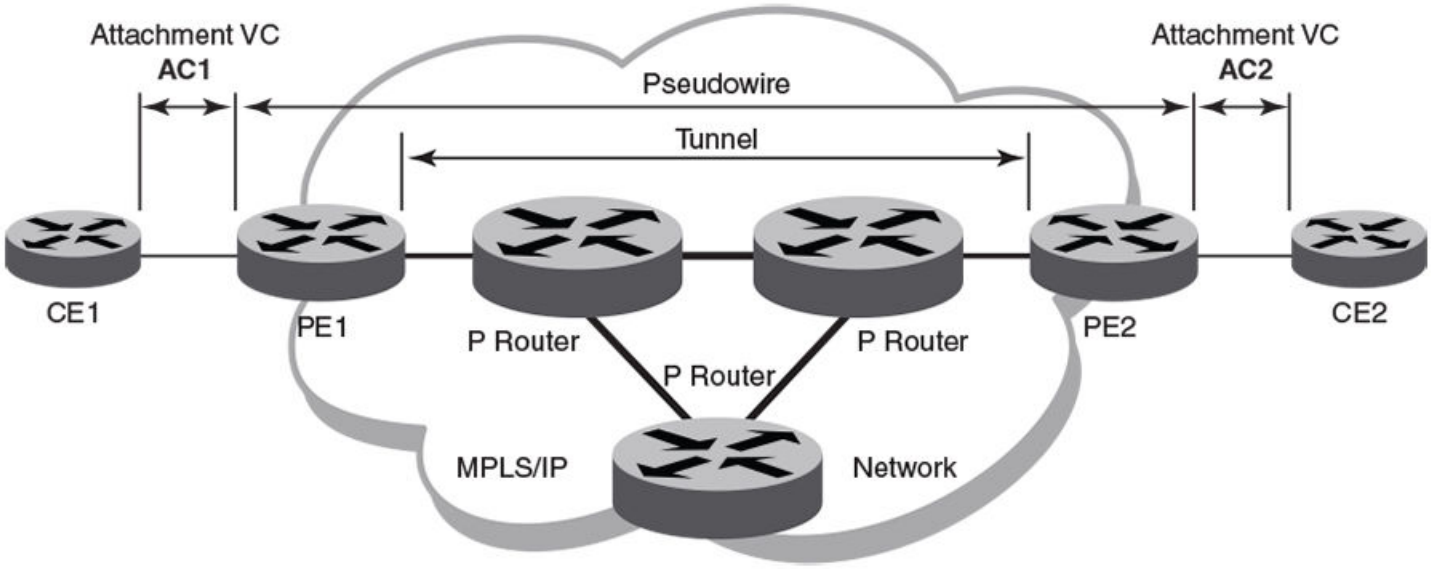
Because it emulates LAN switching, VPLS is considered to be a L2 service that operates over Layer 3 (L3) clouds.

VPLS provides point-to-multipoint (p2mp) functionality while VLL is a special type of VPLS deployment that performs point-to-point (p2p) switching.

VLL is a special type of VPLS deployment that performs point-to-point switching.

The following figure shows a VPLS topology in which switched packets traverse a network.

FIGURE 27 VPLS topology with switching between attachment circuits (ACs) and network core



AC1 and AC2 represent L2 connectivity between customer edge (CE) and provider edge (PE) devices.

Pseudowire is a circuit emulation infrastructure that extends L2 connectivity from CE1 to CE2 by way of PE1 and PE2. The tunnel is typically a L3 tunnel on which a L2 circuit is emulated.

In the case of a packet flowing from CE1 to CE2, the packet enters PE1 from CE1 after the forwarding database (FDB) is used to determine the destination MAC address. Then, a virtual connection (VC) label is imposed prior to encapsulation with the tunnel forwarding information, and the packet is sent out onto the wire towards the network core.

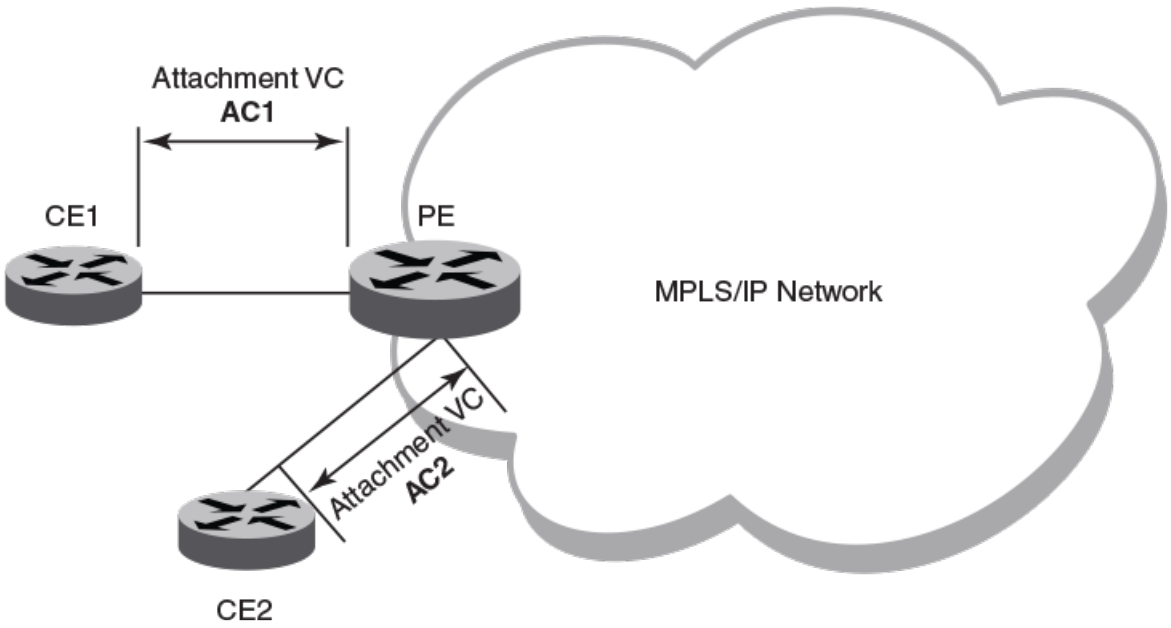
Essentially, the topology in the preceding figure shows a L2 VPN enabling the transport of L2 traffic between two or more native Ethernet networks through an underlying Multiprotocol Label Switching (MPLS) provider network. Customer edge (CE) is the last mile and provider edge (PE) is the first mile node for packets transported towards the provider network. The provider intermediary network is an emulated switch (LAN) or wire (LINE) to the CE. The attachment circuit (AC) represents the logical link between the CE and PE.

An AC may be a port, IEEE 802.1q or IEEE 802.1ad (QinQ)) for Ethernet VPNs. A pseudowire (PW) or emulated wire is used as a transport mechanism to tunnel frames between PEs. A PW is characterized by a circuit identifier, which identifies the destination PE.

MPLS tunnels and paths are established by using routing protocols. PW circuits are established by using signaling.

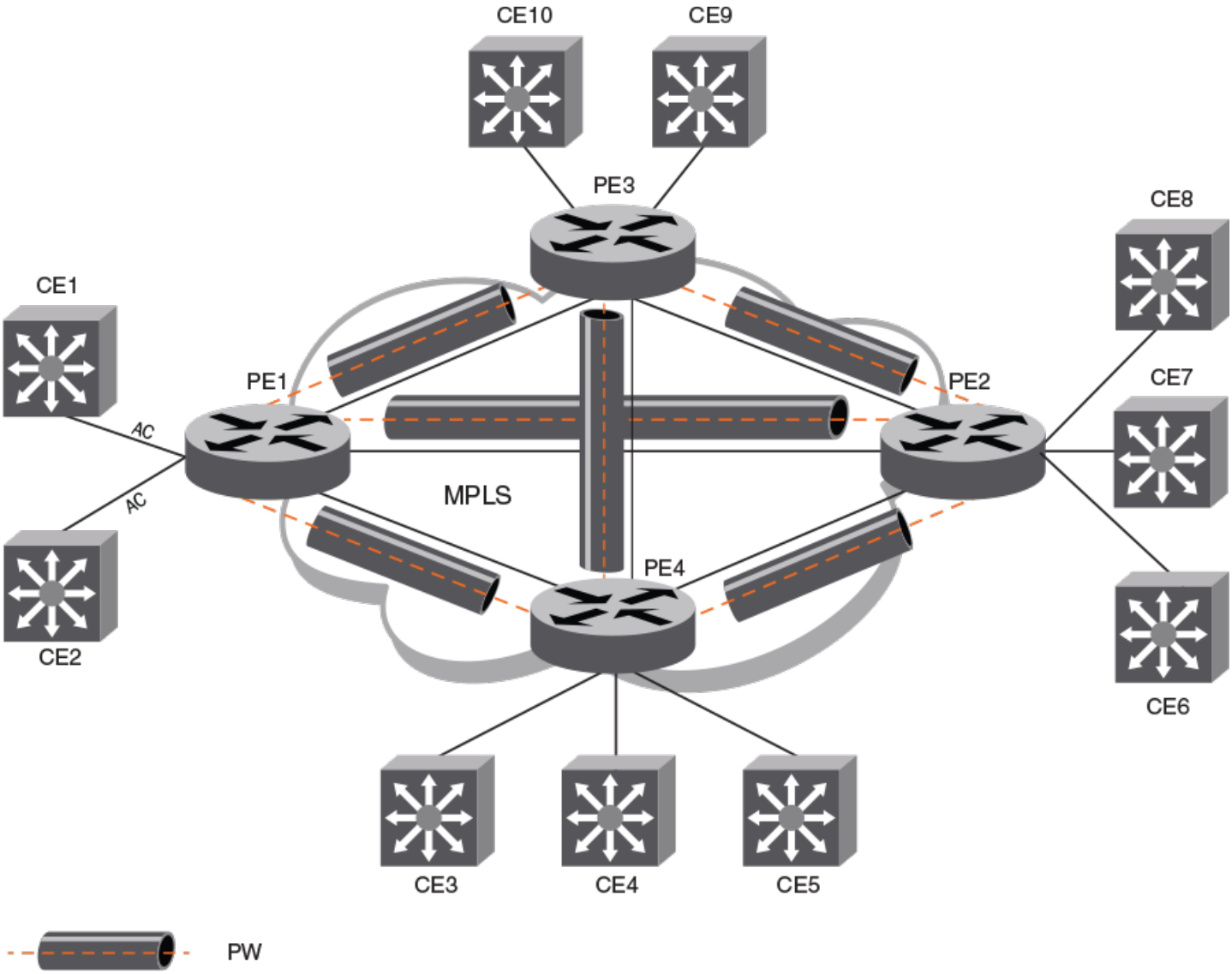
The following figure shows a VPLS topology where switching occurs between two local AC endpoints. This implementation of VPLS does not use VC labels or a pseudowire.

FIGURE 28 VPLS topology with local switching



The following figure shows a common VPLS deployment; an enterprise LAN service. The CE devices represent customer edge devices while the PE devices represent provider edge devices.

FIGURE 29 Enterprise LAN service (VPLS)



VLL

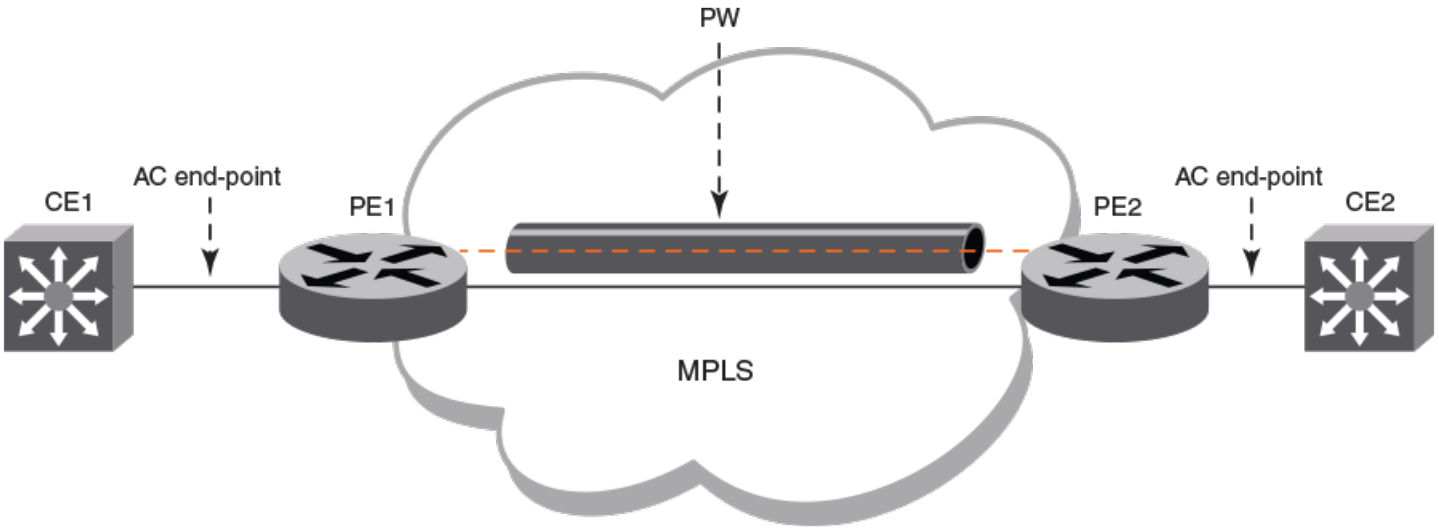
Virtual Leased Line (VLL) is a Layer 2 Virtual Private Network (L2 VPN) architecture that provides point-to-point Ethernet line or Virtual Private Wire Services (VPWS).

A VLL instance is a special type of VPLS deployment.

VLL provides point-to-point (p2p) connectivity between two access networks or endpoints. Typically, a VLL is used to connect two sites that are geographically apart.

The following figure depicts an enterprise VLL service.

FIGURE 30 Enterprise leased line service (VLL)



CE1 and CE2 are the customer edge devices in geographically separate sites.

Pseudowire (PW) is a circuit emulation infrastructure that extends L2 connectivity from CE1 to CE2 by way of PE1 and PE2. The tunnel is typically a L3 tunnel on which a L2 circuit is emulated.

VPLS service endpoints

VPLS supports two types of service-endpoints for VPLS and VLL.

Service endpoints can be categorized as:

- AC endpoints
- PW endpoints

An AC endpoint is a L2 link between a PE device and a CE device. The AC endpoint can be an untagged port, or a tagged port with one or more VLANs. AC endpoints with different VLAN tags can be configured in a single VPLS instance.

A VLL instance interconnects two AC endpoints through a pseudowire, while a VPLS instance forms a full mesh interconnection between multiple AC endpoints through multiple PWs.

The following endpoints are supported:

- port-vlan
- port-vlan-vlan
- PW

Both regular port and port-channel interfaces can be used to form port-vlan, untagged port-vlan, and port-vlan-vlan endpoints.

VPLS service endpoints are represented by logical interfaces (LIFs). By using LIFs, features that apply to regular interfaces, such as QoS, can be applied to VPLS service endpoints.

Local switching

The forwarding behavior of AC endpoints in a VPLS instance is controlled by the switching-mode configuration for local endpoints.

When local switching is enabled, traffic is switched and flooded among AC endpoints in addition to between ACs and PWs. When local switching is disabled, the flooding between AC endpoints is suppressed.

When an unknown unicast packet is received on an AC endpoint and local switching is enabled, the packet is flooded to all other AC endpoints and PW endpoints in the VPLS instance. When local switching is disabled, the unknown packet is only flooded to the PW endpoints in the domain.

Regardless of the local switching configuration, an unknown unicast packet that is received on a PW endpoint is flooded to all AC endpoints.

By default, local switching is enabled.

In a VPLS instance that does not have a PW peer and where all endpoints are AC endpoints (Local VPLS), local switching must be enabled.

To avoid receipt of traffic with different VLAN tags on local endpoints, it is recommended that local switching is disabled in a bridge domain where the PW profile is configured with the VC mode option of **raw-passthrough**. Raw passthorugh mode is designed to forward packets between two VPLS peer devices and is not intended for use with local switching.

Pseudowires

A pseudowire (PW) is a virtual circuit (VC) formed between two provider edge (PE) devices that connects peering Virtual Private LAN Services (VPLS) PE nodes.

An Ethernet pseudowire is logically viewed as an L2 nexthop (VC label) that is reachable through an L3 nexthop (LDP label).

The frames from an AC endpoint packet are sent through an ingress pseudowire interface (which abstracts the transport path and packet encapsulations) towards the remote PE. An egress pseudowire interface then abstracts the packet received from a remote PE and hands it over to the corresponding AC end-point.

A pseudowire interface is unidirectional.

PWs support the following underlying MPLS tunnels:

- LDP – Single Path LSP
- RSVP – Single Path LSP
- RSVP – Pri/Sec (Act LSP)
- RSVP – Pri/Sec (Pas LSP)
- FRR: Adaptive LSP (Make Before Break)
- FRR: Protected & detour (1:1)

PWs do not support the following underlying MPLS tunnels:

- FRR: Protected & Bypass (N:1)
- LDP – Multipath LSP (ECMP)
- LDP over RSVP

Pseudowire operation

The pseudowire setup process establishes the reachability of VPLS bridge domain endpoints across an IP or MPLS cloud.

A pseudowire is operational when the following conditions are met:

- VC signaling is established.
- The L3 reachability of the PW peer is resolved.
- At least one AC endpoint within the bridge domain is up.

A pseudowire is non-operational when the following conditions are met:

- No logical interface is configured for the VPLS instance.
- All AC endpoints are non-operational.

Supported pseudowire features

Pseudowires (PWs) support the following features for each configured PW:

- LSP Load Balancing—Load balancing across a maximum of 16 underlying MPLS tunnels.
- Assigned LSP—A maximum of 32 LSPs can be assigned.
- Specific COS—The underlying MPLS tunnel with the closest CoS value is selected for the transport
- Raw, raw-passthrough, or tagged mode—Can be configured by way of the PW profile that is associated with the bridge domain.
- MTU and MTU check—Can be configured by way of the PW profile that is associated with the bridge domain.
- Uniform and pipe mode for QoS
- Statistics—Egress and ingress statistics are supported but must be enabled in the bridge-domain configuration by using the **statistics** command

Unsupported pseudowire features

Pseudowires (PWs) do not support the following features:

- Auto-discovery of peers
- PW redundancy
- Static PW peers
- VC MAC withdraw
- Status TLV update
- VEoVPLS
- OAM
- Multicast snooping
- Extended counters
- High availability—Process restart
- High availability—ISSU

VLAN tag manipulation (vc-mode) on pseudowires

The virtual connection (VC) mode configuration for a pseudowire (PW) profile determines how VLAN tags are manipulated when a packet is received or transmitted on the PW.

The following table describes VC modes that are supported on PWs.

TABLE 23 VC modes supported on pseudowires

VC mode	Description
Raw	At VC label imposition, when a tagged packet is received on a tagged AC endpoint, the VLAN tag is removed before it is sent out on the wire. When an untagged packet is received on an untagged AC endpoint it is encapsulated as is and sent out on the wire.
Raw-passthrough	Enables interoperation with third-party devices. When a packet that is destined for a remote peer is received on either a tagged or untagged AC endpoint, it is encapsulated in an MPLS header and sent on to the MPLS cloud without adding or removing VLAN tags. When a packet that is destined for a local endpoint is received on either a tagged or untagged AC endpoint, the MPLS header is removed before sending it on to the local endpoint; VLAN tags in the original packet are not changed in any way.
Tagged	At VC label imposition, when a tagged packet is received on a tagged AC endpoint, the packet is encapsulated as is and sent out on the wire. When a packet is from a dual-tagged AC endpoint, the outer VLAN tag is removed and only the inner VLAN TAG is sent out on the wire. When an untagged packet is received on an untagged AC endpoint, a dummy tag is added and it is sent out on the wire.

The VC mode is agreed by PE peer devices during the pseudowire signaling process.

A single VPLS instance can have a mixture of tagged and untagged endpoints.

When the VC mode is changed on a device, the PWs are torn down are re-established except in the cases of a change from raw to raw-passthrough or a change from raw-passthrough to raw. The traffic impact is minimal (because PWs are not torn down and re-established) when the VC mode is changed from raw to raw-passthrough (or vice versa).

VC mode is configured by specifying the **vc-mode** option for the **pw-profile** command.

NOTE
When a pseudowire profile is attached to a bridge domain, on which routing is enabled (by using the **router-interface** command), you are not allowed to change the pseudowire profile **vc-mode** configuration to **raw**.

PW statistics

Pseudowire (PW) statistics are supported for both the ingress (packet going out over the PW) and egress (packet received on the VC-Label of the PW) provider edge (PE) devices.

PW statistics are enabled or disabled per bridge domain and apply to all the PWs that are part of the bridge domain. The logical interfaces for inbound and outbound statistics are shared resources. Hence, the corresponding PW is operational only when these hardware resources are available.

PW statistics are configured using the **statistics** command in bridge domain configuration mode. For further information about enabling statistics on a bridge domain, refer to *Bridge Domains*.

Pseudowire control word and flow label

Pseudowire (PW) control word and flow label improve PW traffic flow over a Multi-Protocol Label Switched (MPLS) packet switched network (PSN).

- PW control word supports optimal transport of Ethernet Protocol Data Units (PDUs) over an MPLS PSN.
- PW flow label supports improved load balancing of PW traffic over an MPLS PSN.

PW control word

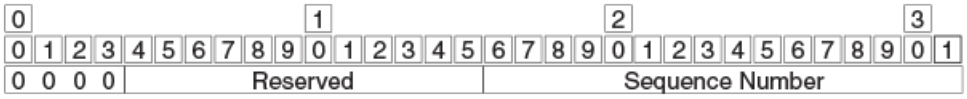
To ensure that a PW operates correctly over an MPLS PSN, the PW traffic is normally transported over a single network path, even when equal-cost multiple-paths (ECMPs) exist between the ingress and egress PW provider edge (PE) devices. Transporting PW traffic over a single network prevents misordered packet delivery to the egress PE device.

Because the standard MPLS label stack does not contain an explicit protocol identifier, label switching routers (LSRs) in ECMP implementations may examine the first nibble after the MPLS label stack to determine if a labeled packet is an IP packet. When the source MAC address of an Ethernet frame that is carried over a PW (without a control word present) begins with 0x4 or 0x6, it could be mistaken for an IPv4 or IPv6 packet. Depending on the configuration and topology of the MPLS network, this misinterpretation could lead to a situation in which all packets for a specific PW do not follow the same path and may increase out-of-order frames on the specific PW, or cause packets to follow a different path than actual traffic.

PW control word (RFC 4385) enables all packets for a specific PW to follow the same path over an MPLS PSN.

The following figure shows the format of the PW control word.

FIGURE 31 PW control word format



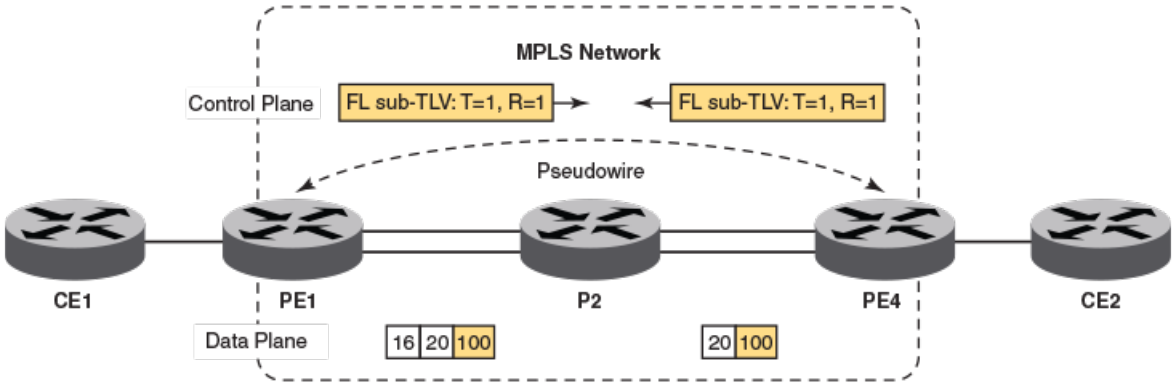
The first 4 bits are set to 0 to indicate PW data and distinguish a PW packet from an IP packet. The next 12 bits are reserved.

The last 16 bits carry a PW-specific sequence number that guarantees ordered frame delivery. A sequence number value of 0 indicates that the sequence number check algorithm is not used.

> **NOTE**
> PW control word is a nonconfigurable, global system value.

The following figure shows the operation of PW control word in a PW over MPLS network configuration.

FIGURE 32 PW over MPLS



- The ingress device (PE1) initiates the PW setup, including the capability to transmit and receive PW control word by using the new interface parameter TLV.
- The egress device (PE4) signals a control word-handling capability.
- When PE1 receives traffic from the customer edge device (CE1), it pushes the label stack onto the traffic control word and forwards the packet to P2.
- P2 uses the field immediately after the bottom of the MPLS label stack to identify the payload as non-IP, IPv4 or IPv6 and forwards the packet to PE4.
- PE4 receives the packet with two labels; the PW label, which identifies the PW, and PW control word, which is discarded without processing.

Pseudowire (PW) control word is configured by enabling control word for either a PW profile or for PW-related devices in a bridge domain. To enable control word for a PW profile, use the **control-word** command in PW profile configuration mode. To enable control word for PW-related devices in a bridge domain, use the **peer** command in bridge domain configuration mode.

PW flow label

Some PWs transport high volumes of traffic that comprise multiple separate traffic flows (for example, all packets for the same source-destination pair for a Transport Control Protocol [TCP] connection is a specific traffic flow).

To avoid latency and jitter, it is important that packets for a specific traffic flow are mapped to the same links along the path to the egress device.

PWs that carry multiple traffic flows require only the preservation of packet ordering in the context of an individual traffic flow and do not require the preservation of packet order for all traffic carried on the PW.

In normal cases, routers use source and destination IP addresses, protocol type, and TCP or UDP port numbers as keys in a load-balancing algorithm to find the appropriate outgoing interface. In MPLS networks, deep packet inspection may be needed for LSRs to identify such keys.

PW flow label is a mechanism that supports load balancing in an MPLS network, without the need for deep packet inspection by LSRs.

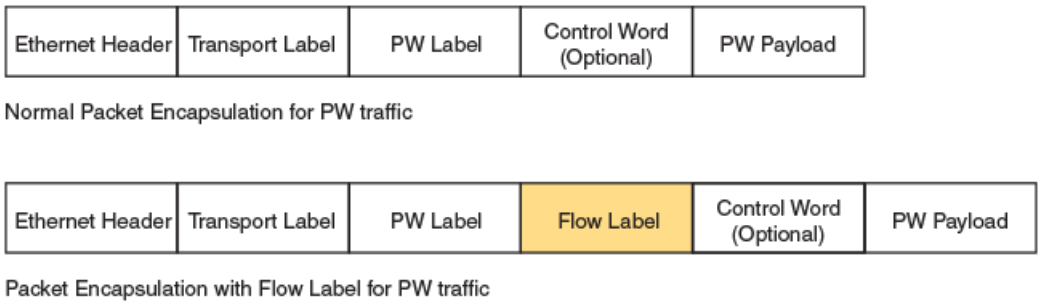
The PW flow label is added to the bottom of the label stack, by the ingress LSR (which has complete information about the packet) before it pushes the label stack. Transit LSRs can use the entire label stack (including the flow label) as a key for a load-balancing algorithm.

The PW flow label allows PWs to leverage the availability of, for example, equal-cost multiple-path (ECMP) between LSRs in an MPLS PSN. The ingress PE device maps individual traffic flows within a PW to the same flow label. Label Distribution Protocol (LDP) uses the PW flow label to map packets for a specific traffic flow to the same links along the path and to ensure that packets for the specific traffic flow follow the same path over the MPLS PSN. The PW flow label is discarded at the egress PE device.

> NOTE
> PW flow label load balancing is not supported when the incoming packet contains more than three labels.

The following figure shows packet encapsulation when flow label is enabled for PW traffic.

FIGURE 33 PW flow label

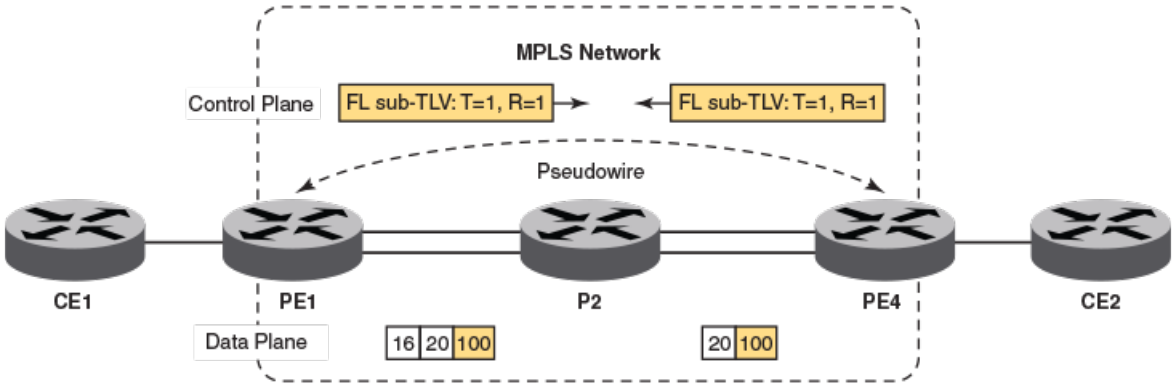


PW flow label is not a reserved value (that is, in the range from 0 through 15). To prevent any processing of flow label at the egress LSR, the TTL value of the flow label is set to 0.

> NOTE
> PW flow label is not signaled between PE routers. The flow label is generated locally and pushed to the bottom of the stack by the ingress LSR.

The following figure, which shows a PW over MPLS network configuration, describes the operation of PW flow label.

FIGURE 34 PW over MPLS



- The ingress device (PE1) initiates the PW setup, including the capability to transmit and receive PW flow label by using the new interface parameter TLV.
- The egress device (PE4) signals a flow label-handling capability.
- When PE1 receives traffic from the customer edge router, it uses a hash algorithm based on the keys (destination MAC address, source MAC address, and so on) to generate a flow label. It then pushes the label stack onto the flow label (S=1, TTL=0) and forwards the packet to P2.
- The P2 router uses the bottom label from the stack to ensure that a particular traffic flow (packets for a specific source-destination pair) follows the same path through transit routers.
- PE4 receives the packet with two labels with the top label being the PW label, which is used to identify the pseudowire, and the flow label, which is discarded without processing.

  NOTE
  When load balancing is based only on PW flow label, the transit device (P2) must be configured to include the bottom-of-stack (BOS) label by using the **lag hash bos start** command.

Pseudowire (PW) flow label is configured either by enabling flow label either for a PW profile or the bridge domain to which the PW is attached.

Pseudowire (PW) flow label is configured by enabling flow label for either a PW profile or PW-related devices in a bridge domain. To enable flow label for a PW profile, use the **flow-label** command in PW profile configuration mode. To enable flow label for PW-related devices in a bridge domain, use the **peer** command in bridge domain configuration mode.

Supported VPLS features

The following VPLS features are supported:

- Local VPLS (without PWs)
- VPLS—untagged, single-tagged, and dual-tagged endpoints
- Flooding of L2 BPDUs in VPLS
- VPLS tagged, raw, and raw-passthrough modes for virtual circuits
- Dynamic LAG support for VPLS endpoints
- VPLS MTU enforcement

- VPLS static MAC address support for AC endpoints
- VPLS over Multi-Chassis Trunking (MCT)
- Virtual Ethernet (VE) over VPLS

Configuration of VPLS and VLL

Configuration of a VPLS or VLL instance includes configuring a bridge-domain, configuring a virtual connection (VC) identifier, configuring logical interfaces for attachment circuit (AC) endpoints, and configuring peer IP addresses.

To configure a VPLS or VLL instance, you must complete the following tasks:

- Configure a bridge domain.
- Configure a VC identifier.
- Configure logical interfaces for AC endpoints.
- (Optional) Configure a pseudowire (PW) profile.
- Configure peer IP addresses. Configuring peer IP addresses creates PW endpoints.

NOTE
VPLS (or VLL) configuration is separate from the underlying IP or MPLS configuration. MPLS tunnels need to be brought up separately. For further information about the configuration of MPLS tunnels, refer to the *Extreme SLX-OS MPLS Configuration Guide* for the SLX 9850 Router.

QoS treatment in VPLS packet flow

There are default behaviors for Quality of Service (QoS) propagation in VPLS forwarding on PE routers.

On the ingress label-edge router (LER), the final EXP value for the VC label is not dependant on the CoS value in the VC-peer configuration.

By default, for traffic flowing from a CE device to a PE device, 3 bits of the PCP field from the incoming Ethernet frame header are extracted and mapped to an internal CoS value by way of an ingress CoS map. This internal value is then mapped to an outgoing CoS value by way of an egress CoS map. The outgoing CoS value is then inserted into the EXP field in the outgoing VC label. When incoming traffic does not have VLAN tag, the default PCP value that is configured on a port is used.

In the case of traffic received from the network core side, by default the EXP field from the incoming VC label is mapped to an internal CoS value by way of an ingress CoS map. This internal value is then mapped to an outgoing CoS value by way of an egress CoS map. The outgoing CoS value is then inserted into the PCP field in the Ethernet frame header going out to the CE device.

On the egress LER, the CoS value for the VC-peer configuration is not dependant on the final EXP value for the VC label.

The following table shows ingress and egress behavior for different global, tunnel and PW configuration combinations.

TABLE 24 Ingress and Egress LER behavior

Global	Tunnel	PW	AC to PW (per path)	PW to AC (per PW)
Uniform	No CoS	No CoS	Uniform	Uniform
Uniform	CoS	No CoS	Pipe	Uniform
Uniform	No CoS	CoS	Uniform	Pipe
Uniform	CoS	CoS	Pipe	Pipe
Pipe	No CoS	No CoS	Pipe	Pipe
Pipe	CoS	No CoS	Pipe	Pipe

TABLE 24 Ingress and Egress LER behavior (continued)

Global	Tunnel	PW	AC to PW (per path)	PW to AC (per PW)
Pipe	No CoS	CoS	Pipe	Pipe
Pipe	CoS	CoS	Pipe	Pipe

Configuring a PW profile

A pseudowire (PW) emulates a point-to-point connection over a packet-switching network. PW profile configuration defines PW attributes. After configuration, a PW profile must be attached to a bridge domain.

A pseudowire profile can be shared across multiple bridge domains. Complete the following task to configure a PW profile.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Create a PW profile and enter configuration mode for the profile.

```
device(config)# pw-profile pw_example
```

3. Configure the virtual connection mode for the profile.

```
device(config-pw-pw_example)# vc-mode tag
```

   In this example, tag mode is configured for the PW profile, pw_example.

4. Configure a maximum transmission unit (MTU) of 1300 for the PW profile.

```
device(config-pw-pw_example)# mtu 1300
```

5. Enforce an MTU check during PW signaling.

```
device(config-pw-pw_example)# mtu-enforce true
```

The following example creates a PW profile named pw_example and configures attributes for the profile.

```
device# configure terminal
device(config)# pw-profile pw_example
device(config-pw-pw_example)# vc-mode tag
device(config-pw-pw_example)# mtu 1300
device(config-pw-pw_example)# mtu-enforce true
```

Attaching a PW profile to a bridge domain

A pseudowire (PW) profile is applied by attaching it to a bridge domain.

Before it is attached to a bridge domain, the PW profile must be configured.

Perform the following task to attach a PW profile to a bridge domain.

1. From privileged EXEC mode, enter the global configuration mode.

```
device# configure terminal
```

2. Enter bridge domain configuration mode.

```
device(config)# bridge-domain 501
```

3. Configure a PW profile for the bridge domain.

```
device(config-bridge-domain-501)# pw-profile pw_example
```

4. Return to privileged EXEC mode.

```
device(config-bridge-domain-501)# end
```

5. Verify the configuration.

```
device# show bridge-domain 501

Bridge-domain 501
-------------------------------
Bridge-domain Type: MP , VC-ID: 501
Number of configured end-points:  2 , Number of Active end-points: 2
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
PW-profile: pw_example, mac-limit: 0
VLAN: 501, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth1/6.501
Un-tagged Ports:
Total VPLS peers: 1 (1 Operational):

VC id: 501, Peer address: 10.9.9.9, State: Operational, uptime: 4 sec
      Load-balance: True , Cos Enabled: False,
      Tunnel cnt: 4
      rsvp  p101(cos_enable:False cos_value:0)
      rsvp  p102(cos_enable:False cos_value:0)
      rsvp  p103(cos_enable:False cos_value:0)
      rsvp  p104(cos_enable:False cos_value:0)
      Assigned LSPs count:4 Assigned LSPs:p101 p102 p103 p104
      Local VC lbl: 989041, Remote VC lbl: 983040,
      Local VC MTU: 1500, Remote VC MTU: 1500,
      Local VC-Type: 5, Remote VC-Type: 5
```

The following example shows how to attach a PW profile named pw_example to bridge domain 501.

```
device# configure terminal
device(config)# bridge-domain 501
device(config-bridge-domain-501)# pw-profile pw_example
```

Configuring control word for a PW profile

Configuring control word for a pseudowire (PW) profile enables all PW packets to follow the same path over an MPLS network.

Before completing the following task, the PW profile on which control word is to be enabled must be configured. An example is provided at the end of this task that shows all the steps in order.

PW control word configuration enables all packets for a specific PW to follow the same path over the MPLS network supporting, for example, the optimal transport of Ethernet Protocol Data Units (PDUs) over the network.

Perform the following task to configure PW control word for a PW profile.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter PW profile configuration mode

```
device(config)# pw-profile pw_example
```

3. Enable control word for the PW profile.

```
device(config-pw-profile-pw_example)# control-word
```

NOTE
When control word is enabled for a previously configured PW, control word capabilities between PE devices are activated only after LDP neighbors are cleared by using the **clear mpls ldp neighbor** command. For further information on clearing LDP neighbors, refer to *Extreme SLX-OS MPLS Configuration Guide*.

The following example shows how to configure a PW profile and enable control word for the profile. It then shows how to attach the PW profile to a bridge domain.

```
device# device# configure terminal
device(config)# pw-profile pw_example
device(config-pw-pw_example)# vc-mode tag
device(config-pw-pw_example)# mtu 1300
device(config-pw-pw_example)# mtu-enforce true
device(config-pw-pw_example)# control-word
device(config-pw-pw_example)# exit
!
device(config)# bridge-domain 502
device(config-bridge-domain-502)# pw-profile pw_example
```

Configuring PW control word on a bridge domain

Configuring control word for pseudowire-related peer devices in a bridge domain enables all pseudowire (PW) packets to follow the same path over an MPLS network.

Before completing the following task, the relevant bridge domain must be configured. An example is provided at the end of this task that shows all the steps in order.

PW control word configuration enables all packets for a specific PW to follow the same path over the MPLS network supporting, for example, the optimal transport of Ethernet Protocol Data Units (PDUs) over the network.

Perform the following task to configure PW control word for a peer device in a bridge domain.

1. Enable control word for the PW profile.

```
device# configure terminal
```

a) From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

b) Enter bridge domain configuration mode. The following example shows how to enter configuration mode for bridge domain 502.

```
device(config)# bridge-domain 502
```

c) Enable control word for a PW-related peer device (10.9.9.9) that is configured on the bridge domain.

```
device(config-bridge-domain-502)# peer 10.9.9.9 control-word
```

2. Return to privileged EXEC mode.

```
device(config-pw-profile-pw_example)# end
```

3. Verify the configuration.

```
show bridge-domain 502

Bridge-domain 502
-----------------------------
Bridge-domain Type: MP, VC-ID: 502 MCT Enabled: FALSE
Number of configured end-points:  3, Number of Active end-points: 1
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
MAC Withdrawal: Disabled
PW-profile: default, mac-limit: 0
VLAN: 502, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth0/5.502
Un-tagged Ports:
Total VPLS peers: 2 (0 Operational):
VC id: 502, Peer address: 10.9.9.9, State: Wait for LSP tunnel to Peer, uptime: -
Load-balance: True, Cos Enabled: False,
Tunnel cnt: 0
Assigned LSPs count:2 Assigned LSPs:q502 q752
Local VC lbl: 0, Remote VC lbl: 0,
Local VC MTU: 1500, Remote VC MTU: 0,
Local VC-Type: 5, Remote VC-Type: 0
Local PW preferential Status: Active, Remote PW preferential Status: Active
Local Flow Label Tx: Enabled, Local Flow Label Rx: Enabled
Remote Flow Label Tx: Enabled, Remote Flow Label Rx: Enabled
Local Control Word: Enabled, Remote Control Word: Enabled
```

The following example shows how to configure a bridge domain on which control word is enabled for a PW-related peer.

```
device(config)# bridge-domain 502
device(config-bridge-domain-502)# vc-id 8
device(config-bridge-domain-502)# logical-interface ethernet 1/5.15
device(config-bridge-domain-502)# logical-interface port-channel 2.200
device(config-bridge-domain-502)# peer 10.9.9.9 load-balance control-word
device(config-bridge-domain-502)# peer 10.12.12.12 control-word
device(config-bridge-domain-502)# peer 10.12.12.12 lsp lsp1 lsp2
device(config-bridge-domain-502)# local-switching
device(config-bridge-domain-502)# bpdu-drop-enable
device(config-bridge-domain-502)#
device(config-pw-pw_example)# exit
```

Configuring flow label for a PW profile

Configuring flow label for a pseudowire (PW) profile improves load balancing of PW traffic over an MPLS network.

Before completing the following task, the PW profile on which flow label is to be enabled must be configured. An example is provided at the end of this task that shows all the steps in order.

Flow label configuration improves load balancing of PW traffic over an MPLS network, particularly in the context of PWs that transport high volumes of traffic that are comprised of multiple individual traffic flows (for example, the same source-destination pair for a Transport Control Protocol (TCP) connection is an individual traffic flow).

Perform the following task to configure flow label for a PW profile.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter PW profile configuration mode.

```
device(config)# pw-profile pw_example
```

3. Enable flow label for the PW profile.

```
device(config-pw-profile-pw_example)# flow-label
```

The following example shows how to configure a PW profile and enable flow label on the profile. It then shows how to attach the PW profile to a bridge domain.

```
device# device# configure terminal
device(config)# pw-profile pw_example
device(config-pw-pw_example)# vc-mode tag
device(config-pw-pw_example)# mtu 1300
device(config-pw-pw_example)# mtu-enforce true
device(config-pw-pw_example)# flow-label
device(config-pw-pw_example)# exit

device(config)# bridge-domain 5
device(config-bridge-domain-5)# pw-profile pw_example
```

Configuring PW flow label on a bridge domain

Configuring flow label for pseudowire-related peer devices in a bridge domain improves load balancing of pseudowire (PW) traffic over an MPLS network.

Prior to completing the following task, the bridge domain on which flow label is to be enabled must be configured. There is an example at the end of this task that shows all the steps in order.

Flow label configuration improves load balancing of PW traffic over an MPLS network, particularly in the context of PWs that transport high volumes of traffic that are comprised of multiple individual traffic flows (for example, the same source-destination pair for a Transport Control Protocol (TCP) connection is an individual traffic flow).

Perform the following task to configure PW flow label on a bridge domain.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter bridge domain configuration mode. The following example shows how to enter configuration mode for bridge domain 502.

```
device(config)# bridge-domain 502
```

3. Enable flow label for the PW-related peer devices that are configured on the bridge domain.

```
device(config-bridge-domain-502)# peer 10.9.9.9 flow-label
```

4. Verify the configuration.

```
show bridge-domain 502

Bridge-domain 502
-----------------------------
Bridge-domain Type: MP, VC-ID: 502 MCT Enabled: FALSE
Number of configured end-points:  3, Number of Active end-points: 1
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
MAC Withdrawal: Disabled
PW-profile: default, mac-limit: 0
VLAN: 502, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth0/5.502
Un-tagged Ports:
Total VPLS peers: 2 (0 Operational):
VC id: 502, Peer address: 10.9.9.9, State: Wait for LSP tunnel to Peer, uptime: -
Load-balance: True, Cos Enabled: False,
Tunnel cnt: 0
Assigned LSPs count:2 Assigned LSPs:q502 q752
Local VC lbl: 0, Remote VC lbl: 0,
Local VC MTU: 1500, Remote VC MTU: 0,
Local VC-Type: 5, Remote VC-Type: 0
Local PW preferential Status: Active, Remote PW preferential Status: Active
Local Flow Label Tx: Enabled, Local Flow Label Rx: Enabled
Remote Flow Label Tx: Enabled, Remote Flow Label Rx: Enabled
Local Control Word: Enabled, Remote Control Word: Enabled
```

The following example shows how to configure a bridge domain on which flow label is enabled for the PW-related peers.

```
device# configure terminal
device(config)# bridge-domain 5
device(config-bridge-domain-5)# vc-id 8
device(config-bridge-domain-5)# logical-interface ethernet 1/5.15
device(config-bridge-domain-5)# logical-interface port-channel 2.200
device(config-bridge-domain-5)# peer 10.15.15.15 load-balance flow-label
device(config-bridge-domain-5)# peer 10.12.12.12 flow-label
device(config-bridge-domain-5)# peer 10.12.12.12 lsp lsp1 lsp2
device(config-bridge-domain-5)# local-switching
device(config-bridge-domain-5)# bpdu-drop-enable
device(config-bridge-domain-5)#
device(config-pw-pw_example)# exit
```

Configuring a static MAC address over an endpoint in a VPLS instance

A static MAC address can be associated with the logical interface for an attachment circuit (AC) endpoint in a bridge domain.

You can configure a MAC address for a logical interface for an endpoint in a VPLS instance by completing the following task.

NOTE
Pre-configuration for the static mac is supported. Pre-configured static mac is shown as inactive mac.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Create the logical-interface (LIF) entry associated to a physical or port-channel interface, associate the LIF entry to the VPLS instance and configure the static mac associated with the LIF entry.

```
device(config)# mac-address-table static 0011.2345.6789 forward logical-interface ethernet 1/2.200
```

3. Enter privileged EXEC mode.

```
device(config)# exit
```

4. (Optional) Verify the configuration using any of the following commands.

```
device# show mac-address-table
device# show mac-address-table static
device# show mac-address-table bridge-domain [bd-id]
```

Displaying MAC address information for VPLS bridge domains

Various show commands can be used to display MAC address information for bridge domains.

- Enter the **show mac-address-table bridge-domain** command to display information about MAC addresses in VPLS bridge domains. The following example shows details of all MAC addresses learned on all bridge domains.

```
device# show mac-address-table bridge-domain all

VlanId/BD-Id   Mac-address           Type     State     Ports/LIF/peer-ip
629(B)            0011.2222.5555     Dynamic  Active    eth 1/3.100
629(B)            0011.2222.6666     Dynamic  Inactive  eth 1/1.500
629(B)            0011.2222.1122     Dynamic  Active    10.12.12.12
629(B)            0011.2222.3333     static   Inactive  po 5.700
629(B)            0011.0101.5555     Dynamic  Active    eth 1/2.400

Total MAC addresses : 5
```

- Enter the **show mac-address-table bridge-domain** command specifying a bridge domain to display information about MAC addresses for a specific bridge domain.

```
device# show mac-address-table bridge-domain 1

BD-Name   Mac-address     Type     State     Ports/LIF/Peer-IP
1         0011.2222.5555  Dynamic  Active    eth 1/3.200
1         0011.2222.6666  Dynamic  Inactive  eth 2/2.500

Total MAC addresses : 2
```

Configuring a VPLS instance

A virtual private LAN Service (VPLS) instance provides multipoint LAN services.

Prior to completing the following task, the underlying L3 configuration of MPLS tunnels must be completed. In addition, the logical interfaces to be attached the VPLS instance (bridge domain) should be configured. For information on configuring logical interfaces, refer to *Logical Interfaces*.

You can configure a VPLS instance by completing the following task.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Create a multipoint bridge domain.

```
device(config)# bridge-domain 5
```

   By default, the bridge-domain service type is multipoint. In this example, bridge domain 5 is configured as a multipoint service.

3. Configure a virtual connection identifier for the bridge domain.

```
device(config-bridge-domain-5)# vc-id 8
```

4. **NOTE**
   Logical interfaces representing bridge-domain endpoints must be created before they can be bound to a bridge domain.

   Bind the logical interfaces for attachment circuit endpoints to the bridge domain.

```
device(config-bridge-domain-5)# logical-interface ethernet 1/6.400
```

   In this example, Ethernet logical interface 1/6.400 is bound to bridge domain 5.

5. Repeat Step 4 to bind other logical interfaces for attachment circuit endpoints to the bridge domain.

```
device(config-bridge-domain-5)# logical-interface port-channel 2.200
```

   In this example, port channel logical interface 2.200 is bound to bridge domain 5.

6. Configure peer IP addresses to create pseudowire (PW) endpoints.

```
device(config-bridge-domain-5)# peer 10.15.15.15 load-balance
```

   In this example, a peer IP address of 10.15.15.15 is configured under bridge domain 5 and specifies load balancing.

7. Repeat Step 6 to configure more peer IP addresses to create PW endpoints.

```
device(config-bridge-domain-5)# peer 10.12.12.12 lsp lsp1 lsp2
```

   In this example, a peer IP address of 10.12.12.12 under bridge domain 5 and specifies two label-switched paths (lsp1 and lsp2).

8. (Optional) Configure local switching for bridge domain 5.

```
device(config-bridge-domain-5)# local-switching
```

9. (Optional) Enable dropping L2 bridge protocol data units (BPDUs) for bridge domain 5.

```
device(config-bridge-domain-5)# bpdu-drop-enable
```

10. (Optional) Configure a PW profile under the bridge domain 5.

```
device(config-bridge-domain-5)# pw-profile 2
```

The following example creates bridge domain 5 and configures virtual connection identifier 8 for the bridge domain. It binds ethernet and port-channel logical interfaces to the bridge domain and configures peer IP addresses under the domain. It configures local switching, enables dropping of L2 BPDUs, and configures a PW profile for the domain.

```
device# configure terminal
device(config)# bridge-domain 5
device(config-bridge-domain-5)# vc-id 8
device(config-bridge-domain-5)# logical-interface ethernet 1/6.400
device(config-bridge-domain-5)# logical-interface port-channel 2.200
device(config-bridge-domain-5)# peer 10.15.15.15 load-balance
device(config-bridge-domain-5)# peer 10.12.12.12 lsp lsp1 lsp2
device(config-bridge-domain-5)# local-switching
device(config-bridge-domain-5)# bpdu-drop-enable
device(config-bridge-domain-5)# pw-profile 2
```

Configuring a VLL instance

A virtual leased line (VLL) instance provides point-to-point (peer) LAN services.

Prior to completing the following task, the Ethernet logical interface and pseudowire profiles must be created. There is an example at the end of this task that shows all the steps in order.

You can configure a VLL instance by completing the following task.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a point-to-point bridge domain to use VLL services.

   ```
   device(config)# bridge-domain 3 p2p
   ```

   In this example, bridge domain 3 is created as a point-to-point service. By default, the bridge-domain service type is multipoint.

3. Configure a virtual connection identifier for the bridge domain.

   ```
   device(config-bridge-domain-3)# vc-id 500
   ```

4. Bind the logical interfaces for attachment circuit endpoints to the bridge domain.

   ```
   device(config-bridge-domain-3)# logical-interface ethernet 1/5.15
   ```

   In this example, the Ethernet logical interface 1/5.15 is bound to bridge domain 3.

5. Configure peer IP addresses to create pseudowire (PW) endpoints.

   ```
   device(config-bridge-domain-3)# peer 10.10.10.10
   ```

6. Configure a PW profile under the bridge domain.

   ```
   device(config-bridge-domain-3)# pw-profile to-mpls-nw
   ```

   The following example configures a PW profile 2 under bridge domain 3.

7. Repeat this configuration on the other peer device with appropriate parameters.
8. Enter Privileged EXEC mode.

   ```
   device(config-bridge-domain-3)# end
   ```

9. (Optional) Display information about the configured VLL instance.

```
device# show bridge-domain 3

Bridge-domain Type: P2P , VC-ID: 3
Number of configured end-points:2 , Number of Active end-points: 2
VE if-indx: 0, Local switching: FALSE, bpdu-drop-enable: FALSE
PW-profile: default, mac-limit: 0
VLAN: 3, Tagged ports: 1(1 up), Un-tagged ports: 0 (0 up)
Tagged Ports: eth1/5.15
Un-tagged Ports:
Total VLL peers: 1 (1 Operational):
VC id: 3, Peer address: 10.10.10.10, State: Operational, uptime: 18 sec
Load-balance: True , Cos Enabled: False,
Tunnel cnt: 4
rsvp p105 (cos_enable:Falsecos_value:0)
rsvp p106 (cos_enable:Falsecos_value:0)
rsvp p107 (cos_enable:Falsecos_value:0)
rsvp p108 (cos_enable:Falsecos_value:0)
Assigned LSPs count:4 Assigned LSPs:p105 p106 p107 p108
Local VC lbl: 851968, Remote VC lbl: 985331,
Local VC MTU: 1600, Remote VC MTU: 1500,
Local VC-Type: 5, Remote VC-Type: 5
```

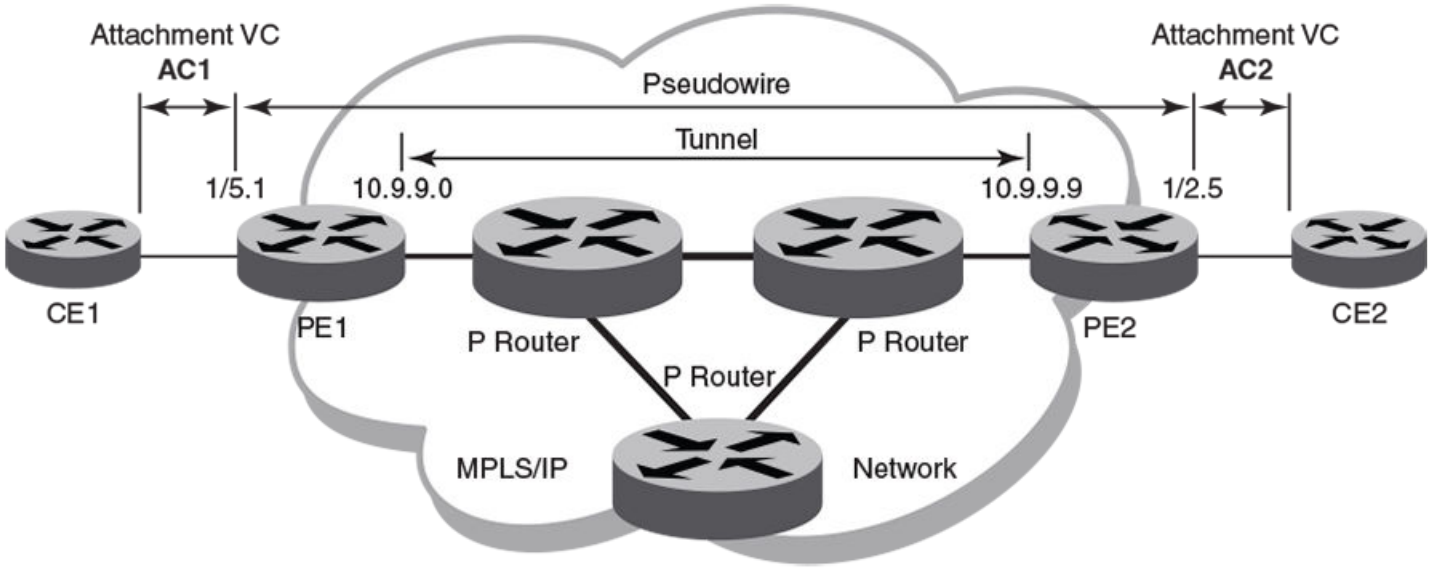
The following example shows the creation of a logical interface and a pseudowire profile in addition to the bridge domain and VLL instance configuration.

```
device# configure terminal
device(config)# interface ethernet 1/5
device(conf-if-eth-1/5)# switchport
device(conf-if-eth-1/5)# switchport mode trunk
device(conf-if-eth-1/5)# switchport trunk tag native-vlan
device(conf-if-eth-1/5)# shutdown
device(conf-if-eth-1/5)# logical-interface ethernet 1/5.15
device(conf-if-eth-lif-1/5.15)# vlan 200
device(conf-if-eth-lif-1/5.15)# exit
device(conf-if-eth-1/5)# exit
device(config)# pw-profile to-mpls-nw
device(config-pw-profile-to-mpls-nw)# mtu 1600
device(config-pw-profile-to-mpls-nw)# mtu-enforce true vc-mode tag
device(config-pw-profile-to-mpls-nw)# exit
device(config)# bridge-domain 3 p2p
device(config-bridge-domain-3)# vc-id 500
device(config-bridge-domain-3)# logical-interface ethernet 1/5.15
device(config-bridge-domain-3)# peer 10.10.10.10
device(config-bridge-domain-5)# pw-profile to-mpls-nw
```

Configuration example for VPLS with switching between ACs and network core

VPLS can be configured with switching between attachment circuits (ACs) and the network core. Because VPLS emulates LAN switching, it is considered to be a Layer 2 (L2) service that operates over Layer 3 (L3) clouds.

FIGURE 35 VPLS configuration with switching between attachment circuits (ACs) and network core



The topology in the preceding figure shows a L2 VPN that enables transport of L2 traffic between two or more native Ethernet networks through an underlying Multiprotocol Label Switching (MPLS) provider network. Customer edge (CE) is the last mile and provider edge (PE) is the first mile node for packets transported towards the provider network. The provider intermediary network is an emulated switch (LAN) or wire (LINE) to the CE. The AC represents the logical link between the CE and PE.

Pseudowire is a circuit emulation infrastructure that extends L2 connectivity from CE1 to CE2 by way of PE1 and PE2. The tunnel is typically a L3 tunnel on which a L2 circuit is emulated.

The following examples show how to configure the provider edge devices (PE1 and PE2) shown in this topology.

PE1

```
device# configure terminal
device(config)# bridge-domain 500 p2mp
device(config-bridge-domain-500)# vc-id 501
device(config-bridge-domain-500)# peer 10.9.9.9 load-balance
device(config-bridge-domain-500)# logical-interface ethernet 1/5.1     ! AC1
device(config-bridge-domain-500)# exit

device(config)# pw-profile default
```

PE2

```
device# configure terminal
device(config)# bridge-domain 300 p2mp
device(config-bridge-domain-300)# vc-id 501
```

```
device(config-bridge-domain-300)# peer 10.9.9.0 load-balance
device(config-bridge-domain-500)# logical-interface ethernet 1/2.5     ! AC2
device(config-bridge-domain-500)# exit

device(config)# pw-profile default
```

VPLS MAC withdrawal

VPLS MAC address withdrawal removes MAC addresses that have been dynamically learned, thus providing faster convergence.

A MAC address withdrawal message is send with a MAC list Type Length Value (TLV). 200 MAC addresses are bulked and sent in one Mac TLV message.

> NOTE
> The MAC withdrawal support is only for explicit MAC addresses in MAC withdrawal TLV. Empty MAC list as well as sending MAC withdrawal TLV to specific subset of peers will not be supported.

The maximum number of MAC addresses supported is 5000 in a 5 second interval. The remaining MAC in the AC LIF are not sent. After the 5 second interval, another LIF down event triggers MAC withdrawal message for a new 5 second interval. MAC withdrawal is supported for both VPLS and MCT-VPLS. MPLS signals the MAC withdraw TLV to all the peers.

The **mac-address withdrawal** command enables MAC withdrawal on the bridge domain. The **no** form of the command disables MAC withdrawal.

Disabling MAC withdrawal on a bridge domain stops sending of MAC withdraw messages. MAC withdraw messages are received at the receiver end and MAC flush happens even when MAC address withdrawal is not enabled.

Enabling VPLS MAC address withdrawal

VPLS MAC address withdrawal removes dynamically-learned MAC addresses.

Convergence is faster when VPLS MAC address withdrawal is enabled.

Perform the following task to enable VPLS MAC address withdrawal.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter bridge domain configuration mode. The following example shows how to enter configuration mode for bridge domain 1.

   ```
   device(config)# bridge domain 1
   ```

3. Enable VPLS MAC address withdrawal.

   ```
   device(config-bridge-domain-1)# mac-address withdrawal
   ```

   After it is enabled, use the **no** form of the command to disable MAC address withdrawal.

4. Return to privileged EXEC mode.

   ```
   device(config-bridge-domain-1)# end
   ```

5. Verify the configuration.

```
device# show bridge-domain
Bridge-domain 1
------------------------------
Bridge-domain Type: MP , VC-ID: 0
Number of configured end-points:  0 , Number of Active end-points: 0
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
MAC Withdrawal: Enabled
PW-profile: default, mac-limit: 0
Total VPLS peers: 0 (0 Operational):
device#
```

The following example shows how to enable VPLS MAC address withdrawal and then verify the configuration for bridge domain 1.

```
device# configure terminal
device(config)# bridge domain 1
device(config-bridge-domain-1)# mac-address withdrawal
device(config-bridge-domain-1)# end
!
device# show bridge-domain
Bridge-domain 1
------------------------------
Bridge-domain Type: MP , VC-ID: 0
Number of configured end-points:  0 , Number of Active end-points: 0
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
MAC Withdrawal: Enabled
PW-profile: default, mac-limit: 0
Total VPLS peers: 0 (0 Operational):
device#
```

Routing over VPLS

Routing over VPLS is known as VEoVPLS and is enabled by configuring a Virtual Ethernet (VE) interface on a Virtual Private LAN Service (VPLS) instance.

> NOTE
> VEoVPLS is not supported for high availability (HA) configurations.

A VPLS instance on which a VE interface is configured can participate in routing protocols (for example, OSPF and ISIS) and exchange routes with provider edge (PE) devices on remote customer sites.

In a VPLS configuration:

- Provider core devices are not aware of customer MAC addresses, VC labels, or VPLS instances
- Provider edge devices:
  - Learn the MAC addresses of locally connected customer devices (attachment circuit [AC] toward customer edge devices [CEs])
  - Flood broadcast and unknown unicast frames to other PE devices in the virtual private network (VPN)
  - Create associations between remote MAC addresses and remote PEs
  - Encapsulate or decapsulate Layer 2 packets with VC and MPLS labels, for sending packets through PWs or toward AC endpoints

By configuring a VE interface on a VPLS instance, VE routing packets can arrive on the VPLS endpoint or from an uplink port. VEoVPLS also routes packets between the VPLS VE interface and all other IP interfaces outside the VPLS domain that reside on the PE device including:

- Physical interfaces

- Other VLAN-based VE interfaces for both tagged and untagged ports
- VE interfaces that reside on other VPLS instances

Multiple VPLS instances may belong to the same VRF instance; for example, you may configure different VPLS instances to different sites while hall sites are in the same routing area.

A unicast packet that is to be routed has the PE MAC address in the MAC destination address field and is subjected to Layer 3 processing.

Routing for VPLS packets is the same as routing for non-VPLS packets. The next-hop address obtained from the routing table may be a VLAN interface, a VPLS endpoint, or a VPLS uplink port.

The following types typify of packet routing on a VPLS instance:

- Packet from the local endpoint of a VPLS VE interface to:
  - A VPLS uplink port of the same VPLS instance or another VPLS instance
  - A local VPLS endpoint of the same VPLS instance or another VPLS instance
  - A VE over VLAN interface
  - A normal IP interface over a physical interface
  - An IPoMPLS interface
  - A GRE tunnel
  - A VXLAN tunnel
- Packet from the remote endpoint (uplink port) of a VPLS VE interface to:
  - A VPLS uplink port of another VPLS instance
  - A local endpoint of the same VPLS instance or another VPLS instance
  - A VE over VLAN interface
  - A normal IP interface over a physical interface
  - An IPoMPLS interface
- Packet from a non-VPLS (VE over VLAN, IPoMPLS or GRE tunnel) interface to a local endpoint on the VPLS instance
- Packet from a non-VPLS (VE over VLAN or IPoMPLS) interface to a remote endpoint on a VPLS instance

VE over VPLS:

- Operates with routing over GRE, VXLAN, MCT, L3 VPN, and VRRP
- Supports both IPv4 and IPv6 addressing
- Supports different platform types for VPLS PE nodes, including MLX Series platforms

The following table describes the operation of VEoVPLS components.

TABLE 25 VEoVPLS component operation

Component	Operation
Routing	A virtual Ethernet (VE) interface supports most of the existing functionality of a legacy IP interface or a VE over VLAN interface: • When the VE interface is disabled, all packets routed to the interface are dropped. • When the VE interface is not configured over a VPLS instance, all the routed traffic to the interface is forwarded, at the Layer-2 level, by VPLS. This forwarding by VPLS, ensures that traffic to the interface does not break existing routing by way of loopback connections that may already be configured.
Switching	The configuration of VE over VPLS has no impact on VPLS Layer 2 switching functionality.
ARP	As with ARP, VEoVPLS ARP maintains the relationship between the Layer 2 MAC address and the IP address: • ARP entries associated with the VEoVPLS interface are resolved on one member of the VPLS instance; that is, either the local endpoint or remote endpoint (remote VPLS peer). • ARP broadcast goes out through each endpoint member of the VPLS instance.

TABLE 25 VEoVPLS component operation (continued)

Component	Operation
	• Static ARP is supported for VEoVPLS in the same way that it is supported for VE over VLAN • Static MAC addresses for local endpoints are supported. • Static MAC addresses for remote endpoints are not supported. • ARP Guard is not supported.
ARP DAI	Dynamic ARP inspection is not supported for VEoVPLS.
ECMP	When a VE interface is configured over a label-switched path (LSP) with a load-balanced-enabled VPLS interface and the next hop is the pseudowire (PW), ECMP is automatically handled; the Layer 3 FEC points to the PW FEC and interface traffic is load-balanced across multiple LSPs.
RPF	• Loose mode RPF is supported. • Strict mode RPF is not supported when the packet is coming from the PW tunnel.
TTL handling	When the next hop is the PW, TTL value handling for VEoVPLS is the same as MPLS TTL behavior; two TTL modes exist: • pipe • uniform (default value) In uniform mode, the TTL value of the IP packet is decremented and copied to the MPLS header. MPLS TTL is decremented at each hop. At the LER router, the TTL value is decremented and copied to the IP header before computing the IP checksum. In pipe mode, the TTL value of the IP packet is decremented and a constant TTL value of 255 is set in the MPLS header. The TTL value of the MPLS packet is decremented at each hop. In the LER router, the MPLS header TTL value is discarded and the inner IP packet TTL value is decremented again before computing the IP checksum. Traceroute ICMP packets work isimilarly to other IP packets; that is, when the TTL value is 0 or 1, the packets are sent, in order, to the CPU for sending the TTL Expiry ICMP packet to the source.
VRF	VEoVPLS is supported on all VRFs. Routing is supported between both remote and local endpoints of a VPLS VE instance and other non-VPLS IP interfaces. All IP interfaces must be in the same VRF. Inter-VRF routing is supported based on leaking routes between VRFs.

OAR for VEoVPLS

One Armed Router (OAR) supports the configuration of multiple logical interfaces (Multi LIF) on a port.

By default, OAR is supported for VPLS AC endpoints.

When Multi LIF support is not required on AC endpoints in a VPLS instance, you can disable it by using the **disallow-oar-ac** command in router interface configuration mode. Disabling OAR on VPLS instances when it is not required helps system scaling of hardware resources.

Enabling routing on a VPLS instance

A Virtual Private LAN Service (VPLS) instance can participate in routing protocols when a virtual Ethernet (VE) interface is enabled.

Before completing this task, configure the VPLS instance on which routing is to be enabled. An example is provided at the end of this task that shows all the configuration steps in order.

NOTE

You cannot enable routing on a VPLS instance when the **vc-mode** option on the PW attached to the instance is set to **raw**.

Perform the following steps to enable routing on a VPLS instance.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter configuration mode for the VPLS instance bridge domain.

```
device(config)# bridge-domain 5
```

   A bridge domain created for a VPLS application is also known as a VPLS instance.

3. Bind a virtual router interface to the bridge domain and enter router interface configuration mode.

```
device(config-bridge-domain-5)# router-interface ve 10
```

   Routing is enabled on the VPLS instance by binding a virtual router interface to the bridge domain that was created for the instance.

4. (Optional) Disallow multiple attachment circuit (AC) endpoints on the virtual router interface.

```
device(config-router-interface-10)# disallow-oar-ac
```

   By default, multiple AC endpoints are allowed on a port. To help with system scaling of hardware resources when multiple endpoints are not required, you can disallow them.

5. Return to privileged EXEC mode.

```
device(config-router-interface-10)# end
```

6. Verify the configuration.

```
device# show running-config bridge-domain 5

bridge-domain 5 p2mp
 vc-id 8
 peer 10.2.2.1
 router-interface ve 10
 !
 logical-interface ethernet 0/38.99
 logical-interface ethernet 0/14.99
 pw-profile default
 bpdu-drop-enable
 local-switching
```

The following example first shows how to configure a VPLS instance (a bridge domain that is configured for a VPLS application), then shows how to enable routing on the instance (by binding a virtual router interface to the bridge domain), and finally shows how to disallow multiple AC endpoints on the virtual router interface.

```
device# configure terminal
device(config)# bridge-domain 5
device(config-bridge-domain-5)# vc-id 8
device(config-bridge-domain-5)# logical-interface ethernet 0/38.99
device(config-bridge-domain-5)# logical-interface ethernet 0/14.99
device(config-bridge-domain-5)# peer 10.2.2.1
device(config-bridge-domain-5)# local-switching
device(config-bridge-domain-5)# bpdu-drop-enable
device(config-bridge-domain-5)# pw-profile 2
!
device(config-bridge-domain-5)# router-interface ve 10
device(config-router-interface-10)# disallow-oar-ac
```

Configuration example for enabling routing over VPLS

After creating a virtual Ethernet (VE) interface and configuring a Virtual Private LAN Service (VPLS) instance, the VE interface is bound to the VPLS instance to enable routing over VPLS.

Configure a VE interface

```
device# configure terminal
device(config)# interface ve 99
device(config-if-Ve-99)# ip proxy-arp
device(config-if-Ve-99)# ip address 10.1.1.1/24
device(config-if-Ve-99)# no shutdown
device(config-if-Ve-99)# exit
```

Configure a logical interface

```
device# configure terminal
device(config)# interface Ethernet 1/1
device(conf-if-eth-1/1)# switchport
device(conf-if-eth-1/1)# switchport mode trunk
device(conf-if-eth-1/1)# switchport trunk allowed vlan add 100
device(conf-if-eth-1/1)# switchport trunk tag native-vlan
device(conf-if-eth-1/1)# no shutdown
device(conf-if-eth-1/1)# logical-interface ethernet 1/1.99
device(conf-if-eth-lif-1/1.99)# vlan 99
device(conf-if-eth-lif-1/1.99)# end
```

Configure a PW profile

```
device(config)# pw-profile test1
device(config-pw-test1)# vc-mode tag
device(config-pw-test1)# exit
```

Configure a VPLS instance

```
device(config)# bridge-domain 99 p2mp
device(config-bridge-domain-99)# vc-id 10
device(config-bridge-domain-99)# router-interface ve 99
device(config-bridge-domain-99)# logical-interface ethernet 1/1.99
device(config-bridge-domain-99)# pw-profile test1
device(config-bridge-domain-99)# bpdu-drop-enable
device(config-bridge-domain-99)# local-switching
```

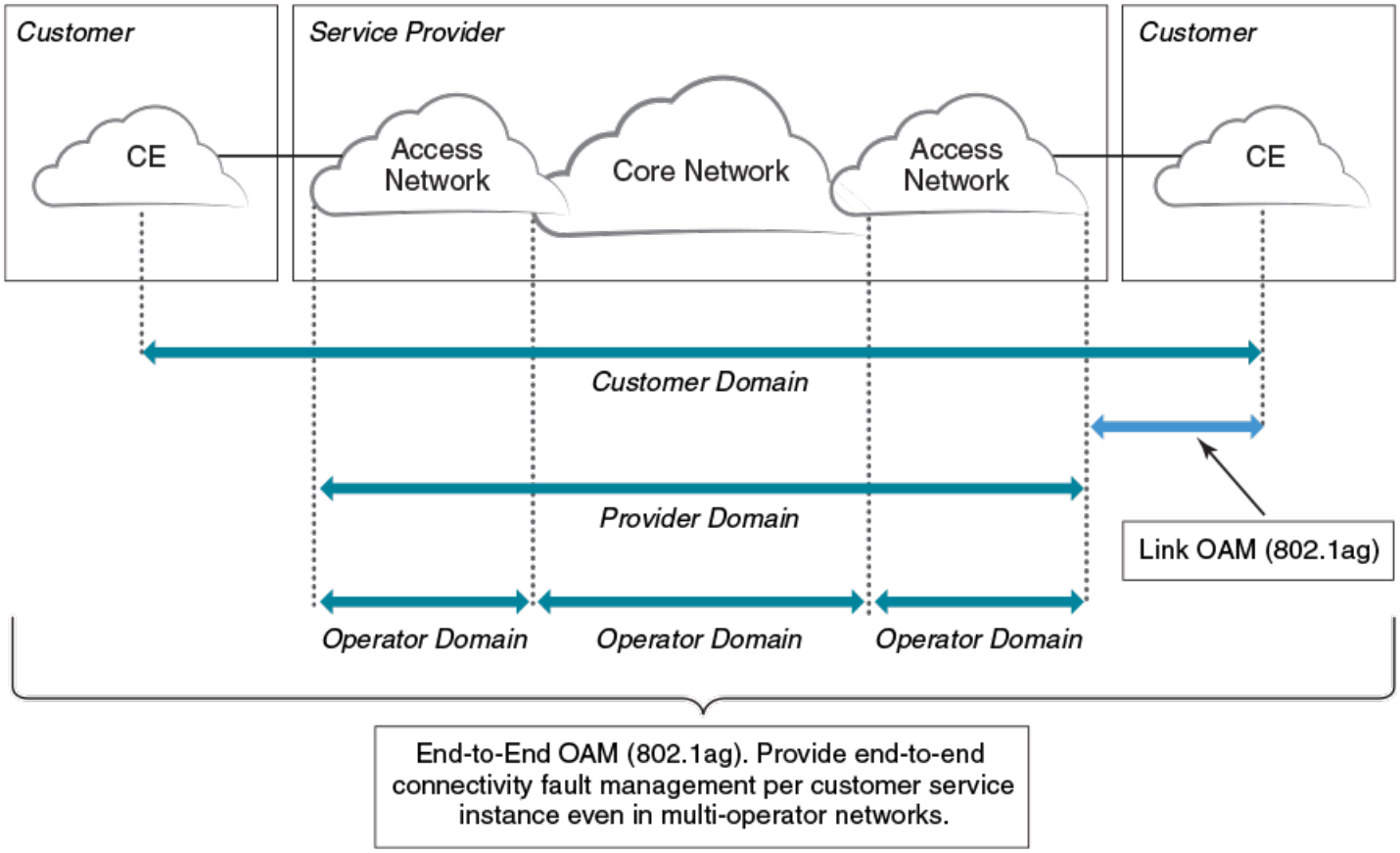
802.1ag Connectivity Fault Management

Bridges are increasingly used in networks operated by multiple independent organizations, each with restricted management access to each other's equipment. CFM provides capabilities for detecting, verifying and isolating connectivity failures in such networks.

There are multiple organizations involved in a Metro Ethernet Service: Customers, Service Providers and Operators.

Customers purchase Ethernet Service from Service Providers. Service Providers may utilize their own networks, or the networks of other Operators to provide connectivity for the requested service. Customers themselves may be Service Providers, for example a Customer may be an Internet Service Provider which sells Internet connectivity.

FIGURE 36 OAM Ethernet tools



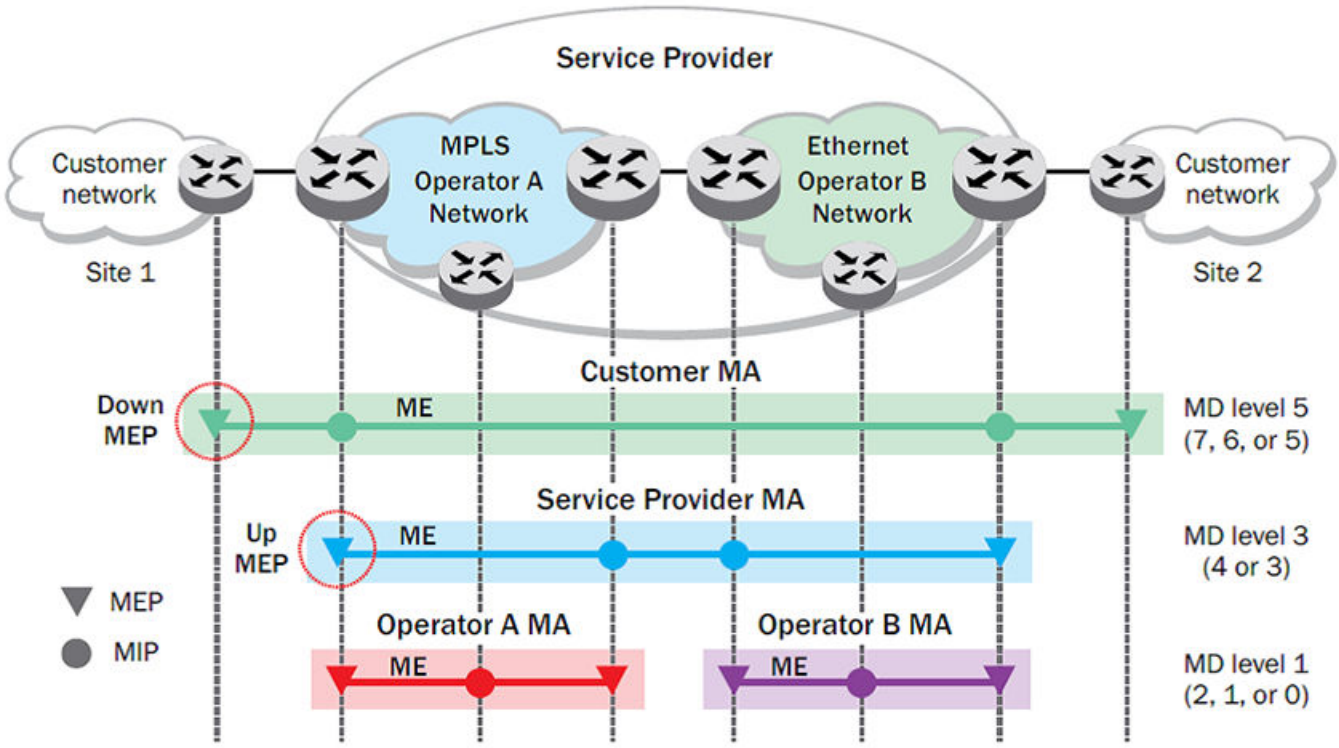
Maintenance Domain (MD)

A Maintenance Domain is part of a network controlled by a single operator. In the following figure, a customer domain, provider domain and operator domain are described.

The Maintenance Domain (MD) levels are carried on all CFM frames to identify different domains. For example, in the following figure, some bridges belong to multiple domains. Each domain associates to an MD level.

- Customer Level: 5-7
- Provider Level: 3-4
- Operator Level: 0-2

FIGURE 37 CFM deployment



Maintenance Association (MA)

Every MD can be further divided into smaller networks having multiple Maintenance End Points (MEP). Usually an MA is associated with a service instance (for example, a VLAN or a VPLS).

Maintenance End Point (MEP)

An MEP is located on the edge of an MA and defines the endpoint of the MA. Each MEP has unique ID (MEPID) within the MA. The connectivity in a MA is defined as connectivity between MEPs. MEPs generate a Continuity Check Messages that are multicast to all other MEPs in same MA to verify the connectivity.

Each MEP has a direction, down or up. Down MEPs receive CFM PDUs from the LAN and sends CFM PDUs towards the LAN. Up MEPs receive CFM PDUs from a bridge relay entity and sends CFM PDUs towards the bridge relay entity on a bridge. End stations support down MEPs only, as they have no bridge relay entities.

Maintenance Intermediate Point (MIP)

An MIP is located within a MA. It responds to Loopback and Linktrace messages for Fault isolation.

CFM Hierarchy

MD levels create a hierarchy in which 802.1ag messages sent by customer, service provider, and operators are processed by MIPs and MEPs at the respective level of the message. A common practice is for the service provider to set up a MIP at the customer MD level at the edge of the network, as shown in the figure above, to allow the customer to check continuity of the Ethernet service to the edge of the network. Similarly, operators set up MIPs at the service provider level at the edge of their respective networks, as shown in the figure above, to allow service providers to check the continuity of the Ethernet service to the edge of the operators' networks. Inside an operator network, all MIPs are at the respective operator level, also shown in the figure above.

Mechanisms of Ethernet IEEE 802.1ag OAM

Mechanisms supported by IEEE 802.1ag include Connectivity Check (CC), Loopback, and Link trace. Connectivity Fault Management allows for end-to-end fault management that is generally reactive (through Loopback and Link trace messages) and connectivity verification that is proactive (through Connectivity Check messages).

Fault detection (continuity check message)

Each MEP transmits periodic multicast CCMs towards other MEPs. For each MEP, there is 1 transmission and n-1 receptions per time period. Each MEP has a remote MEP database. It records the MAC address of remote MEPs.

Fault verification (Loopback messages)

A unicast Loopback Message is used for fault verification. A Loopback message helps a MEP identify the precise fault location along a given MA. A Loopback message is issued by a MEP to a given MIP along an MA. The appropriate MIP in front of the fault responds with a Loopback reply. The MIP behind the fault do not respond. For Loopback to work, the MEP must know the MAC address of the MIP to ping.

Fault isolation (Linktrace messages)

Linktrace mechanism is used to isolate faults at Ethernet MAC layer. Linktrace can be used to isolate a fault associated with a given Virtual Bridge LAN Service. Note that fault isolation in a connectionless (multi-point) environment is more challenging than a connection oriented (point-to-point) environment. In case of Ethernet, fault isolation can be even more challenging since a MAC address can age out when a fault isolates the MAC address. Consequently a network-isolating fault results in erasure of information needed for locating the fault.

Enabling or disabling CFM

To enable or disable the Connectivity Fault Management (CFM) protocol globally on the devices and enter into the CFM protocol configuration mode, enter the following command.

```
device# configure terminal
device(config)# protocol cfm
device(config-cfm)#
```

The **no** form of the command disables the CFM protocol.

Creating a Maintenance Domain

A Maintenance Domain (MD) is the network or the part of the network for which faults in connectivity are to be managed. A Maintenance Domain consists of a set of Domain Service Access Points.

An MD is fully connected internally. A Domain Service Access Point associated with an MD has connectivity to every other Domain Service Access Point in the MD, in the absence of faults.

Each MD can be separately administered.

The **domain-name** command in Connectivity Fault Management (CFM) protocol configuration mode creates a maintenance domain with a specified level, name, and ID and enters the specific MD mode specified in the command argument.

```
device# configure terminal
device(config)# protocol cfm
device(config-cfm)# domain-name md1 id 1 level 4
device(config-cfm-md-md1)#
```

The **no** form of the command removes the specified domain from the CFM protocol configuration mode.

Creating and configuring a Maintenance Association

Perform the following steps to create and configure a Maintenance Association (MA).

1. Create a MA within a specific domain, use the **ma-name** command.

   ```
   device# configure terminal
   device(config)# protocol cfm
   device(config-cfm)# domain name md1 id 1 level 4
   device(config-cfm-md-md1)# ma-name ma1 id 1 vlan-id 30 priority 4
   device(config-cfm-md-ma-ma1)#
   ```

   This command changes the Maintenance Domain (MD) mode to the specific MA mode.

2. Set the time interval between two successive Continuity Check Messages (CCMs) that are sent by Maintenance End Points (MEP) in the specified MA, use the **ccm-interval** command.

   ```
   device# configure terminal
   device(config)# protocol cfm
   device(config-cfm)# domain name md1 id 1  level 4
   device(config-cfm-md-md1)# ma-name ma1 id 1 vlan-id 30 priority 3
   device(config-cfm-md-ma-ma1)# ccm-interval 10-second
   device(config-cfm-md-ma-ma1)#
   ```

   The **id** field specifies the short MAID format that is carried in the CCM frame. The default time interval is 10 seconds.

3. Add local ports as MEP to a specific maintenance association using the **mep** command in MA mode.

   ```
   device# configure terminal
   device(config)# protocol cfm
   device(config-cfm)# domain name md1 id 1 level 4
   device(config-cfm-md-md1)# ma-name ma1 id 1 vlan-id 30 priority 3
   device(config-cfm-md-ma-ma1)# mep 1 down ethernet 1/2
   device(config-cfm-md-ma-mep-1)#
   ```

   To configure a CFM packet to a **Down MEP**, you must sent it out on the port on which it was configured. To configure a Connectivity Fault Management (CFM) packet to an **Up MEP**, you must sent it to the entire VLAN for multicast traffic and the unicast traffic must be sent to a particular port as per the MAC table.

4. Configure the remote MEPs using the **remote-mep** command.

```
device# configure terminal
device(config)# protocol cfm
device(config-cfm)# domain name md1 id 1clevel 4
device(config-cfm-md-md1)# ma-name ma1 id 1 vlan-id 30 priority 3
device(config-cfm-md-ma-ma1)# mep 1 down ethernet 1/2
device(config-cfm-md-ma-mep-1)# remote-mep 2
device(config-cfm-md-ma-mep-1)#
```

   If a remote MEP is not specified, the remote MEP database is built based on the CCM. If one remote MEP never sends CCM, the failure cannot be detected.

5. Configure the conditions to automatically create MIPs on ports using the **mip-policy** command, in Maintenance Association mode.

```
device# configure terminal
device(config)# protocol cfm
device(config-cfm)# domain name md1 id 1 level 4
device(config-cfm-md-md1)#ma-name ma1 id 1 vlan-id 30 pri 7
device(config-cfm-md-ma-ma1)#mip-policy explicit
device(config-cfm-md-ma-ma1)#
```

   A MIP can be created on a port and VLAN, only when explicit or default policy is defined for them. For a specific port and VLAN, a MIP is created at the lowest level. Additionally, the level created should be the immediate higher level than the MEP level defined for this port and VLAN.

Displaying CFM configurations

The following commands are used to display the CFM configurations and connectivity status.

show cfm

Use the **show cfm** command to display the Connectivity Fault Management (CFM) configuration.

```
device# show cfm
Domain: md1
Index: 1
Level: 7
  Maintenance association: ma5
  MA Index: 5
  CCM interval: 100 ms
  Bridge-Domain ID: 50
  Priority: 7
  MAID Format: Short
  MEP   Direction  MAC              PORT       VLAN         INNER-VLAN   PORT-STATUS-TLV
  ====  =========  =========        ====       ====         ===========  ===============
  1     UP         609c.9f5f.700d   Eth 1/9    50           --           N
```

NOTE

For the **show cfm** command to generate output, you must first enable CFM in protocol configuration mode.

show cfm connectivity

Use the **show cfm connectivity** command to display the Connectivity Fault Management (CFM) configuration.

The following commands display the received port status tlv state at RMEP.

```
device# show cfm connectivity
Domain: md1
Index: 1
Level: 7
  Maintenance association: ma5
  MA Index: 5
  CCM interval: 100 ms
  Bridge-Domain ID: 50
  Priority: 7
  MAID Format: Short
    MEP Id: 1
    MEP Port: Eth 1/9
      RMEP  MAC             VLAN/PEER         INNER-VLAN    PORT      STATE
      ====  ===             =========         ==========    ====      =====
      2     609c.9f5e.4809  19.1.1.1           --            --       OK
```

NOTE

For the **show cfm** command to generate output, you must first enable CFM in protocol configuration mode.

show cfm brief

Use the **show cfm brief** command to display the Connectivity Fault Management (CFM) brief output.

```
device# show cfm brief
Domain: md1
Index: 1
Level: 7  Num of MA: 1
  Maintenance association: ma5
  MA Index: 5
  CCM interval: 100 ms
  Bridge-Domain ID: 50
  Priority: 7
  MAID Format: Short
  Num of MEP: 1  Num of RMEP: 1
  rmepfail: 0 rmepok: 1
```

802.1d Spanning Tree Protocol

Spanning Tree Protocol overview

The Spanning Tree Protocol (STP) prevents Layer 2 loops in a network by providing redundant links. If a primary link fails, the backup link is activated and network traffic is not affected. STP also ensures that the least cost path is taken when multiple paths exist between ports or VLANs.

The IEEE 802.1d Spanning Tree Protocol (STP) runs on bridges and switches that are 802.1d-compliant.

These variants are Rapid STP (RSTP), Multiple STP (MSTP), Per-VLAN Spanning Tree Plus (PVST+), and Rapid-PVST+ (R-PVST+)

When the spanning tree algorithm is run, the network switches transform the real network topology into a spanning tree topology. In an STP topology any LAN in the network can be reached from any other LAN through a unique path. The network switches recalculate a new spanning tree topology whenever there is a change to the network topology.

For each LAN, the switches that attach to the LAN select a designated switch that is the closest to the root switch. The designated switch forwards all traffic to and from the LAN. The port on the designated switch that connects to the LAN is called the designated port. The switches decide which of their ports is part of the spanning tree. A port is included in the spanning tree if it is a root port or a designated port.

STP runs one spanning tree instance (unaware of VLANs) and relies on long duration forward-delay timers for port state transition between disabled, blocking, listening, learning and forwarding states.

Spanning Tree Protocol configuration notes

Enabling the Spanning Tree Protocol (STP) creates a loop-free topology of Ethernet LANs connected by bridge devices.

The Extreme device supports STP as described in the IEEE 802.1d-1998 specification.

The STP is disabled by default on the Extreme device. Thus, any new VLANs you configure on the Extreme device have STP disabled by default.

Optional features

The following STP configuration features are optional:

- Root guard
- BPDU guard
- PortFast

STP states

Each Layer 2 interface participating in a spanning tree is in one of five states.

A network topology of bridges typically contains redundant connections to provide alternate paths in case of link failures. The redundant connections create a potential for loops in the system. As there is no concept of time to live (TTL) in Ethernet frames, a situation may arise where there is a permanent circulation of frames when the network contains loops. To prevent this, a spanning tree connecting all the bridges is formed in real time.

Every Layer 2 interface running the STP is in one of these states:

State	Action or inaction
Blocking	The interface does not forward frames. Redundant ports are put in a blocking state and enabled when required. This is a transitional state after initialization.
Listening	The interface is identified by the spanning tree as one that should participate in frame forwarding. This is a transitional state after the blocking state for a legacy STP.
Learning	The interface prepares to participate in frame forwarding. This is a transitional state after the blocking state for a legacy STP.
Forwarding	The interface forwards frames. This is a transitional state after the learning state.
Disabled	The interface is not participating in a spanning tree because of shutdown of a port or the port is not operationally up. Any of the other states may transition into this state.

BPDUs

To build a spanning tree for the bridge topology, the bridges must exchange control frames called Bridge Protocol data units (BPDUs).

To construct a spanning tree requires knowledge of the all the participants. The bridges must determine the root bridge and compute the port roles (root, designated, or blocked) with only the information that they have. To ensure that each bridge has enough information, the bridges use BPDUs to exchange information about bridge IDs and root path costs.

A bridge sends a BPDU frame using the unique MAC address of the port itself as a source address, and a destination address of the STP multicast address 01:80:C2:00:00:00.

BPDUs are exchanged regularly (every 2 seconds by default) and enable switches to keep track of network changes and to start and stop forwarding through ports as required.

When a device is first attached to a switch port, it does not immediately forward data. It instead goes through a number of states while it processes inbound BPDUs and determines the topology of the network. When a host is attached, after a listening and learning delay of about 30 seconds, the port always goes into the forwarding state. The time spent in the listening and learning states is determined by the forward delay. However, if instead another switch is connected, the port may remain in blocking mode if it would cause a loop in the network.

There are four types of BPDUs in the original STP specification:

- Configuration BPDU (CBPDU) is used for spanning tree computation.
- Topology Change Notification (TCN) BPDU is used to announce changes in the network topology.
- RSTP BPDU is used for RSTP
- MSTP BPDU is used for MSTP

TCN BPDUs

TCN BPDUs are used to inform other switches of port changes.

TCNs are injected into the network by a non-root switch and propagated to the root. Upon receipt of the TCN, the root switch will set a Topology Change flag in its normal BPDUs. This flag is propagated to all other switches to instruct them to rapidly age out their forwarding table entries.

Consider these configuration rules:

- TCN BPDUs are sent per VLAN.
- TCN BPDUs are sent only in those VLANs in which a topology change is detected.
- TCN BPDUs are sent only in those VLANs for which the bridge is not the root bridge.
- If a topology change is detected on a VLAN for which the bridge is the root bridge, the topology change flag is set in the configuration BPDU that is sent out.

For a given link, in conjunction with the configuration rules, a TCN BPDU is sent out as follows:

- On an access port, only a standard IEEE TCN BPDU is sent out. This TCN BPDU corresponds to a topology change in the access VLAN.
- On a trunk port, if VLAN 1 is allowed (either untagged or untagged), a standard IEEE TCN BPDU is sent for VLAN 1.
- On a trunk port, if the native VLAN is not 1, an untagged TCN BPDU is sent to Cisco or Extreme proprietary MAC address for that VLAN.
- On a trunk port, a tagged TCN BPDU is sent to Cisco or Extreme proprietary MAC address for a tagged VLAN.

As part of the response to TCN BPDUs, the Topology Change and Topology Change Acknowledgment flags are set in all configuration BPDUs corresponding to the VLAN for which the TCN was received.

When a topology change is detected on a trunk port, it is similar to detecting topology changes in each VLAN that is allowed on that trunk port. TCN BPDUs are sent for each VLAN as per the rules.

STP configuration guidelines and restrictions

Follow these configuration guidelines and restrictions when configuring STP and STP variants:

- Only one form of a standing tree protocol, such as STP or RSTP, can be enabled at a time. You must disable one form of xSTP before enabling another.
- When any form of STP is enabled globally, that form of STP is enabled by default on all switch ports.
- LAGs are treated as normal links for any form of STP.
- The STP is disabled by default on the SLX device. Thus, any new VLANs you configure on the SLX device have STP disabled by default.
- PVST/RPVST BPDUs are flooded only if PVST/RPVST is not enabled. STP/RSTP (IEEE) BPDUs are never flooded if STP/RSTP is not enabled.

Understanding the default STP configuration

You should be familiar with STP defaults before you make configuration changes.

TABLE 26 Default STP configuration

Parameter	Default setting
Spanning-tree mode	By default, STP, RSTP, and MSTP are disabled

TABLE 26 Default STP configuration (continued)

Parameter	Default setting
Bridge priority	32768
Bridge forward delay	15 seconds
Bridge maximum aging time	20 seconds
Error disable timeout timer	Disabled
Error disable timeout interval	300 seconds
Port-channel path cost	Standard
Bridge hello time	2 seconds

The following table lists the switch defaults for the interface-specific configuration.

TABLE 27 Default interface specific configuration

Parameter	Default setting
Spanning tree	Enabled on the interface
Automatic edge detection	Disabled
Path cost	2000
Edge port	Disabled
Guard root	Disabled
Hello time	2 seconds
Link type	Point-to-point
Portfast	Disabled
Port priority	128
BPDU restriction	Restriction is disabled.

STP features

The following sections discuss root guard, BPDU guard, and PortFast.

Root guard

Root guard can be used to predetermine a root bridge location and prevent rogue or unwanted switches from becoming the root bridge.

At times it is necessary to protect the root bridge from malicious attack or even unintentional misconfigurations where a bridge device that is not intended to be the root bridge becomes the root bridge, causing severe bottlenecks in the data path. These types of mistakes or attacks can be avoided by configuring root guard on ports of the root bridge.

The root guard feature provides a way to enforce the root bridge placement in the network and allows STP and its variants to interoperate with user network bridges while still maintaining the bridged network topology that the administrator requires. Errors are triggered if any change from the root bridge placement is detected.

When root guard is enabled on a port, it keeps the port in designated FORWARDING state. If the port receives a superior BPDU, which is a root guard violation, it sets the port into a DISCARDING state and triggers a Syslog message and an SNMP trap. No further traffic will be forwarded on this port. This allows the bridge to prevent traffic from being forwarded on ports connected to rogue or wrongly configured STP or RSTP bridges.

Root guard should be configured on all ports where the root bridge should not appear. In this way, the core bridged network can be cut off from the user network by establishing a protective perimeter around it.

Once the port stops receiving superior BPDUs, root guard automatically sets the port back to a FORWARDING state after the timeout period has expired.

BPDU guard

In an STP environment, switches, end stations, and other Layer 2 devices use BPDUs to exchange information that STP will use to determine the best path for data flow.

In a valid configuration, edge port-configured interfaces do not receive BPDUs. If an edge port-configured interface receives a BPDU, an invalid configuration exists, such as the connection of an unauthorized device. The BPDU Guard provides a secure response to invalid configurations because the administrator must manually put the interface back in service.

BPDU guard removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

In some instances, it is unnecessary for a connected device, such as an end station, to initiate or participate in an STP topology change. In this case, you can enable the STP BPDU guard feature on the Extreme device port to which the end station is connected. The STP BPDU guard shuts down the port and puts it into an "error disabled" state. This disables the connected device's ability to initiate or participate in an STP topology. A log message is then generated for a BPDU guard violation, and a message is displayed to warn the network administrator of an invalid configuration.

The BPDU Guard provides a secure response to invalid configurations because the administrator must manually put the interface back in service with the **no shutdown** command if error disable recovery is not enabled by enabling the **errdisable-timeout** command. The interface can also be automatically configured to be enabled after a timeout. However, if the offending BPDUs are still being received, the port is disabled again.

Expected behavior in an interface context

When BPDU Guard is enabled on an interface, the device is expected to put the interface in Error Disabled state when BPDU is received on the port when edge-port and BPDU guard is enabled on the switch interface. When the port ceases to receive the BPDUs, it does not automatically switch to edge port mode, you must configure **error disable timeout** or **no shutdown** on the port to move the port back into edge port mode.

Error disable recovery

A port is placed into an error-disabled state when:

- A BPDU guard violation or loop detection violation occurs
- The number of inError packets exceeds the configured threshold
- An EFM-OAM enabled interface receives a critical event from the remote device (functionally equivalent to a disable state)

Once in an error disable state, the port remains in that state until it is re-enabled automatically or manually.

In STP, RSTP, MSTP, PVST+, or R-PVST+ mode, you can specify the time in seconds it takes for an interface to time out. The range is from 10 through 1000000 seconds. The default is 300 seconds. By default, the timeout feature is disabled.

PortFast

PortFast allows an interface to transition quickly to the forwarding state.

Consider the following when configuring PortFast:

- Do not enable PortFast on ports that connect to other devices.
- PortFast only needs to be enabled on ports that connect to workstations or PCs. Repeat this configuration for every port connected to workstations or PCs.
- Enabling PortFast on ports can cause temporary bridging loops, in both trunking and nontrunking mode.
- If BPDUs are received on a PortFast- enabled interface, the interface loses the edge port status unless it receives a **shutdown**/**no shutdown** command.
- PortFast immediately puts the interface into the forwarding state without having to wait for the standard forward time.

STP parameters

The following section discusses bridge parameters.

Bridge parameters

These parameters are set in STP, RSTP, MSTP, PVST+, and R-PVST+.

Bridge priority

Use this parameter to specify the priority of a device and to determine the root bridge.

Each device has a unique bridge identifier called the bridge ID. The bridge ID is an 8 byte value that is composed of two fields: a 2 B bridge priority field and the 6 B MAC address field. The value for the bridge priority ranges from 0 to 61440 in increments of 4096. The default value for the bridge priority is 32768. You use the **bridge-priority** command to set the appropriate values to designate a device as the root bridge or root device. A default bridge ID may appear as 32768.768e.f805.5800. If the bridge priorities are equal, the device with the lowest MAC address is elected the root.

After you decide what device to designate as the root, you set the appropriate device bridge priorities. The device with the lowest bridge priority becomes the root device. When a device has a bridge priority that is lower than that of all the other devices, it is automatically selected as the root.

The root device should be centrally located and not in a "disruptive" location. Backbone devices typically serve as the root because they usually do not connect to end stations. All other decisions in the network, such as which port to block and which port to put in forwarding mode, are made from the perspective of the root device.

You may also specify the bridge priority for a specific VLAN. If the VLAN parameter is not provided, the priority value is applied globally for all per-VLAN instances. However, for the VLANs that have been configured explicitly, the per-VLAN configuration takes precedence over the global configuration.

Bridge Protocol data units (BPDUs) carry information between devices. All the devices in the Layer 2 network, participating in any variety of STP, gather information on other devices in the network through an exchange of BPDUs. As the result of exchange of the BPDUs, the device with the lowest bridge ID is elected as the root bridge

When setting the bridge forward delay, bridge maximum aging time, and the hello time parameters keep in mind that the following relationship should be kept:

```
(2 × (forward-delay − 1)) ≥ max-age ≥ (2 ×(hello-time + 1))
```

Bridge forward delay

The bridge forward delay parameter specifies how long an interface remains in the listening and learning states before the interface begins forwarding all spanning tree instances. The valid range is from 4 through 30 seconds. The default is 15 seconds.

Additionally, you may specify the forward delay for a specific VLAN. If the VLAN parameter is not provided, the bridge forward delay value is applied globally for all per-VLAN instances. However, for the VLANs that have been configured explicitly, the per-VLAN configuration takes precedence over the global configuration.

Bridge maximum aging time

You can use this setting to configure the maximum length of time that passes before an interface saves its BPDU configuration information.

Keeping with the inequality shown above, when configuring the maximum aging time, you must set the value greater than the hello time. The range of values is 6 through 40 seconds while he default is 20 seconds.

You may specify the maximum aging for a specific VLAN. If the VLAN parameter is not provided, the priority value is applied globally for all per-VLAN instances. However, for the VLANs that have been configured explicitly, the per-VLAN configuration takes precedence over the global configuration.

Bridge hello time

You can use this parameter to set how often the device interface broadcasts hello BPDUs to other devices.

Use the **hello-time** command to configure the bridge hello time. The range is from 1 through 10 seconds. The default is 2 seconds.

You may also specify the hello time for a specific VLAN. If the VLAN parameter is not provided, the priority value is applied globally for all per-VLAN instances. However, for the VLANs that have been configured explicitly, the per-VLAN configuration takes precedence over the global configuration.

Error disable timeout parameter

Configure this parameter to enable a timer that brings a port out of the disabled state.

These parameters are be set in STP, RSTP, MSTP, PVST+, and R-PVST+.

When the STP BPDU guard disables a port, the port remains in the disabled state unless the port is enabled manually. The parameter specifies the time in seconds it takes for an interface to time out. The range is from 10 through 1000000 seconds. The default is 300 seconds.

By default, the timeout feature is disabled.

Port-channel path cost parameter

You configure this parameter to specify the port channel path cost.

This parameter can be set in STP, RSTP, MSTP, PVST+, and R-PVST+ mode.

There are two path cost options:

- Custom - Specifies that the path cost changes according to the port channel bandwidth.
- Standard - Specifies that the path cost does not change according to the port channel bandwidth.

The default port cost is standard.

Configuring STP

The following section discusses configuring STP.

Enabling and configuring STP globally

Follow these steps to enable or disable STP and configure STP parameters.

You can enable STP or STP with one or more parameters enabled.

The parameters can be configured individually by:

1. Entering the commands in steps 1 and 2
2. Running the relevant parameter command
3. Verifying the result
4. Saving the configuration

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable STP globally.

   ```
   device(config)# protocol spanning-tree stp
   ```

   A spanning tree can be disabled by entering the **no protocol spanning-tree stp** command.

3. Describe or name the STP.

   ```
   device(config-stp)# description stp1
   ```

   A description is not required.

4. Specify the bridge priority.

   ```
   device(config-stp)# bridge-priority 4096
   ```

   The bridge with the lowest priority number (highest priority) is designated the root bridge . The range of values is 0 through 61440; values can be set only in increments of 4096. The default priority is 32678.

5. Specify the bridge forward delay.

   ```
   device(config-stp)# forward-delay 20
   ```

   The forward delay specifies how long an interface remains in the listening and learning states before it begins forwarding all spanning tree instances. The valid range is from 4 through 30 seconds. The default is 15 seconds.

6. Configure the maximum aging time.

   ```
   device(config-stp)# max-age 25
   ```

   This parameter controls the maximum length of time that passes before an interface saves its BPDU configuration information. You must set the maximum age to be greater than the hello time. The range is 6 through 40 seconds. The default is 20 seconds.

7. Configure the maximum hello time.

```
device(config-stp)# hello-time 8
```

   The hello time determines how often the switch interface broadcasts hello BPDUs to other devices. The default is 2 seconds while the range is from 1 through 10 seconds.

8. Enable the error disable timeout timer.

```
device(config-stp)# error-disable-timeout enable
```

   This parameter enables a timer that brings a port out of the disabled state. By default, the timeout feature is disabled.

9. Set the error disable timeout timer.

```
device(config-stp)# error-disable-timeout interval 60
```

   When enabled the default is 300 seconds and the range is from 10 through 1000000 seconds.

10. Configure the port channel path cost.

```
device(config-stp)# port-channel path-cost custom
```

   Specifying **custom** means the path cost changes according to the port channel's bandwidth.

11. Return to privileged EXEC mode.

```
device(config-stp)# end
```

12. Verify the configuration.

```
device# show spanning-tree brief

 Spanning-tree Mode: Spanning Tree Protocol

      Root ID      Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

      Bridge ID    Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

 Interface    Role  Sts  Cost       Prio  Link-type     Edge
 ------------------------------------------------------------
 Eth 0/2      DES   FWD  2000       128   P2P            No
 Eth 0/20     DIS   DIS  20000000   128   P2P            No
 Eth 0/25     DIS   DIS  20000000   128   P2P            No
 Eth 0/30     DIS   DIS  20000000   128   P2P            No
 Eth 0/31     DIS   DIS  2000000    128   P2P            No
```

   Observe that the settings comply with the formula set out in the STP parameter configuration section, as:

   `(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))`

   Or in this case `38 ≥ 25 ≥ 18`.

13. Save the configuration.

```
device# copy running-config startup-config
```

STP configuration example

```
device# configure terminal
device(config)# protocol spanning-tree stp
device(config-stp)# description stpForInterface
device(config-stp)# bridge-priority 4096
device(config-stp)# forward-delay 20
device(config-stp)# max-age 25
device(config-stp)# hello-time 8
device(config-stp)# error-disable-timeout enable
device(config-stp)# error-disable-timeout interval 60
device(config-stp)# port-channel path-cost custom
device(config-stp)# end
device# show spanning-tree brief
device# copy running-config startup-config
```

Enabling and configuring STP on an interface

Follow these steps to enable STP and STP features on an interface.

Globally enable STP and STP parameters.

The parameters can be configured individually by:

1. Entering the commands in steps 1-3
2. Running the relevant parameter command
3. Verifying the result
4. Saving the configuration

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter interface configuration mode.

   ```
   device(config)# interface ethernet 0/20
   ```

3. Enable the interface.

   ```
   device(conf-if-eth-0/20)# no shutdown
   ```

4. Configure the path cost for spanning tree calculations on the interface.

   ```
   device(conf-if-eth-0/20)# spanning-tree cost 10000
   ```

   The lower the path cost means a greater chance that the interface becomes the root port. The range is 1 through 200000000. The default path cost is assigned as per the port speed.

5. Enable BPDU guard on the interface.

   ```
   device(conf-if-eth-0/20)# spanning-tree port-fast bpdu-guard
   ```

   BPDU guard removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

6. Configure Root Guard on the interface.

```
device(conf-if-eth-0/20)# spanning-tree guard root
```

   Root Guard protects the root bridge from malicious attacks and unintentional misconfigurations where a bridge device that is not intended to be the root bridge becomes the root bridge.

7. Specify an interface link-type.

```
device(conf-if-eth-0/20)# spanning-tree link-type point-to-point
```

   Specifying a point-to-point link enables rapid spanning tree transitions to the forwarding state. Specifying a shared link disables spanning tree rapid transitions. The default setting is point-to-point.

8. Specify port priority to influence the selection of root or designated ports.

```
device(conf-if-eth-0/20)# spanning-tree priority 64
```

   The range is from 0 through 240 in increments of 16. The default value is 128.

9. Verify the configuration.

```
device# show spanning-tree brief

 Spanning-tree Mode: Spanning Tree Protocol

      Root ID      Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

      Bridge ID    Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

 Interface    Role  Sts  Cost       Prio  Link-type     Edge
 ------------------------------------------------------------
 Eth 0/2      DES   FWD  2000       128   P2P            No
 Eth 0/20     DES   FWD  1000        64   P2P            No
 Eth 0/25     DIS   DIS  20000000   128   P2P            No
 Eth 0/30     DIS   DIS  20000000   128   P2P            No
 Eth 0/31     DIS   DIS  2000000    128   P2P            No
```

   NOTE

   Observe that the settings comply with the formula set out in the STP parameters section, as:
   `(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))`
   or in this case: `38 ≥ 25 ≥ 18`

```
device# show running-config interface ethernet 0/20
interface ethernet 0/20
switchport
switchport mode access
switchport access val 1
spanning-tree cost 1000
spanning-tree guard root
spanning-tree link-type point-to-point
spanning-tree portfast bpdu-guard
spanning-tree priority 64
```

10. Save the settings by copying the running configuration to the startup configuration.

```
device# copy running-config startup-config
```

STP on an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 0/20
device(conf-if-eth-0/20)# no shutdown
device(conf-if-eth-0/20)# spanning-tree cost 10000
device(conf-if-eth-0/20)# spanning-tree port-fast bpdu-guard
device(conf-if-eth-0/20)# spanning-tree guard root
device(conf-if-eth-0/20)# spanning-tree link-type point-to-point
device(conf-if-eth-0/20)# spanning-tree priority 64
device(conf-if-eth-0/20)# end
device# show spanning-tree brief
device# copy running-config startup-config
```

Configuring basic STP parameters

Follow this example to configure basic STP behavior.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable STP globally

   ```
   device(config)# protocol spanning-tree stp
   ```

3. Name the STP.

   ```
   device(config-stp)# description stp1
   ```

4. Designate the root switch.

   ```
   device(conf-stp)# bridge-priority 28672
   ```

   The priority values can be set only in increments of 4096. The range is 0 through 61440.

5. Specify the bridge forward delay.

   ```
   device(config-stp)# forward-delay 20
   ```

6. Configure the maximum aging time.

   ```
   device(config-stp)# max-age 25
   ```

7. Configure the maximum hello time.

   ```
   device(config-stp)# hello-time 8
   ```

8. Enable the error disable timeout timer.

   ```
   device(config-stp)# error-disable-timeout enable
   ```

9. Set the error disable timeout timer interval.

   ```
   device(config-stp)# error-disable-timeout interval 60
   ```

10. Enable port fast on switch ports.
    a) Configure port fast on Ethernet port 0/1.

```
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# spanning-tree portfast
device(conf-if-eth-0/1)# exit
```

    Spanning trees are automatically enabled on switch ports.

    b) Configure port fast on Ethernet port 0/2.

```
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# spanning-tree portfast
device(conf-if-eth-0/2)# exit
```

    c) Repeat these commands for every port connected to workstations or PCs.

```
device(config)# interface ethernet ...
```

11. Specify port priorities to influence the selection of the root and designated ports.

```
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# spanning-tree priority 1
device(conf-if-eth-0/1)# exit
```

12. Enable the guard root feature.

```
device(config)# interface ethernet 0/12
device(conf-if-eth-0/12)# no shutdown
device(conf-if-eth-0/12)# spanning-tree guard root
```

    Root guard lets the device top participate in the STP but only when the device does not attempt to become the root.

13. Return to privileged exec mode.

```
device(conf-if-eth-0/12)# end
```

14. Verify the configuration.

```
device# show spanning-tree brief
Spanning-tree Mode: Spanning Tree Protocol
Root ID Priority 4096
Address 768e.f805.5800
Hello Time 8, Max Age 25, Forward Delay 20
Bridge ID Priority 4096
Address 768e.f805.5800
Hello Time 8, Max Age 25, Forward Delay 20
Interface     Role     Sts     Cost     Prio     Link-type     Edge
-------------------------------------------------------------------
Eth 0/1       DES      FWD     2000     128      P2P           No
Eth 0/2       DES      FWD     2000     128      P2P           No
Eth 0/12      DES      FWD     2000     128      P2P           No
```

    Observe that the settings comply with the formula set out in the STP parameter configuration section, as:

    `(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))`

    or in this case `38 ≥ 25 ≥ 18`.

15. Save the configuration.

```
device# copy running-config startup-config
```

Basic STP configuration example

```
device# configure terminal
device(config)# protocol spanning-tree stp
device(config-stp)# description stp1
device(conf-stp)# bridge-priority 28672
device(config-stp)# forward-delay 20
device(config-stp)# max-age 25
device(config-stp)# hello-time 8
device(config-stp)# error-disable-timeout enable
device(config-stp)# error-disable-timeout interval 60
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# spanning-tree portfast
device(conf-if-eth-0/1)# exit
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# spanning-tree portfast
device(conf-if-eth-0/2)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# spanning-tree priority 1
device(conf-if-eth-0/1)# exit
device(config)# interface ethernet 0/12
device(conf-if-eth-0/12)# no shutdown
device(conf-if-eth-0/12)# spanning-tree guard root
device(conf-if-eth-0/12)# end
device# show spanning-tree brief
device# copy running-config startup-config
```

Re-enabling an error-disabled port automatically

Enable a port to automatically recover from the error-disabled state after the expiration of an error recovery timer.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter STP configuration mode.

   ```
   device(config)# protocol spanning-tree stp
   ```

3. Enable the error-disable-timeout timer.

   ```
   device(config-stp)# error-disable-timeout enable
   ```

4. Set an interval after which port shall be enabled.

   ```
   device(config-stp)# error-disable-timeout interval 60
   ```

   The interval range is from 0 to 1000000 seconds, the default is 300 seconds.

5. Return to privileged EXEC mode.

   ```
   device(config-stp)# end
   ```

6. Verify the configuration.

```
device# show spanning-tree
 Spanning-tree Mode: Spanning Tree Protocol

 Root Id: 8000.768e.f805.5800 (self)
 Bridge Id: 8000.768e.f805.5800

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: enabled
 Bpdu-guard errdisable timeout interval: 60 sec
```

Automatically re-enable an error-disabled port configuration example

```
device# configure terminal
device(config)# protocol spanning-tree stp
device(config-stp)# error-disable-timeout enable
device(config-stp)# error-disable-timeout interval 60
device(config-stp)# end
device# show spanning-tree
```

Clearing spanning tree counters

Follow these steps to clear spanning tree counters on all interfaces or on the specified interface.

1. Clear spanning tree counters on all interfaces.

```
device# clear spanning-tree counter
```

2. Clear spanning tree counters on a specified Ethernet interface.

```
device# clear spanning-tree counter interface ethernet 0/3
```

3. Clear spanning tree counters on a specified port channel interface.

```
device# clear spanning-tree counter interface port-channel 12
```

   Port channel interface numbers range from 1 through 64.

Clearing spanning tree-detected protocols

Follow these steps to restart the protocol migration process.

These commands force a spanning tree renegotiation with neighboring devices on either all interfaces or on a specified interface.

1. Restart the spanning tree migration process on all interfaces.

```
device# clear spanning-tree detected-protocols
```

2. Restart the spanning tree migration process on a specific Ethernet interface.

```
device# clear spanning-tree detected-protocols interface ethernet 0/3
```

3. Restart the spanning tree migration process on a specific port channel interface.

```
device# clear spanning-tree detected-protocols port-channel 12
```

   Port channel interface numbers range from 1 through 64.

Shutting down STP

Follow these steps to o shut down STP either globally, on a specific interface, or a specific VLAN.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Shut down STP.
   - Shut down STP globally and return to privileged EXEC mode.

```
device(config)# protocol spanning-tree stp
device(config-stp)# shutdown
device(config-stp)# end
```

   - Shut down STP on a specific interface and return to privileged EXEC mode.

```
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# spanning-tree shutdown
device(conf-if-eth-1/2)# end
```

   - Shut down STP on a specific VLAN and return to privileged EXEC mode.

```
device(config)# vlan 10
device(config-vlan-10)# spanning-tree shutdown
device(config-vlan-10)# end
```

3. Verify the configuration.

```
device# show spanning-tree
device#
```

4. Save the running configuration to the startup configuration.

```
device# copy running-config startup-config
```

Shut down STP configuration example

```
device# configure terminal
device(config)# vlan 10
device(config-vlan-10)# spanning-tree shutdown
device(config-stp)# end
device# show spanning-tree
device# copy running-config startup-config
```

NOTE

Shutting down STP on a VLAN is used in this example.

802.1w Rapid Spanning Tree Protocol

Rapid Spanning Tree Protocol overview

The RSTP is a way to provide rapid traffic reconvergence for point-to-point links within a few milliseconds (< 500 milliseconds), following the failure of a bridge or bridge port.

The STP (802.1d) standard was designed at a time when recovering connectivity after an outage within a minute or so was considered adequate performance. With the advent of Layer 3 switching in LAN environments, bridging competes with routed solutions where protocols such as OSPF are able to provide an alternate path in less time.

The RSTP can be seen as evolution of STP standard. It provides rapid convergence of connectivity following the failure of bridge, a bridge port or a LAN. It provides rapid convergence of edge ports, new root ports and port connected through point-to-point links. The port, which qualifies for fast convergence, is derived from the duplex mode of a port. A port operating in full-duplex will be assumed to be point-to-point, while a half-duplex port will be considered as a shared port by default. This automatic setting can be overridden by explicit configuration.

RSTP is designed to be compatible and interoperate with the STP. However, the benefit of the RSTP fast convergence is lost when interacting with legacy STP (802.1d) bridges since the RSTP downgrades itself to the STP when it detects a connection to a legacy bridge.

The states for every Layer 2 interface running the RSTP are as follows:

State	Action
Learning	The interface prepares to participate in frame forwarding.
Forwarding	The interface forwards frames.
Discarding	The interface discards frames. Ports in the discarding state do not take part in the active topology and do not learn MAC addresses.

NOTE
The STP disabled, blocking, and listening states are merged into the RSTP discarding state.

The RSTP port roles for the interface are also different. The RSTP differentiates explicitly between the state of the port and the role it plays in the topology. The RSTP uses the root port and designated port roles defined im the STP, but splits the blocked port role into backup port and alternate port roles:

Backup port	Provides a backup for the designated port and can only exist where two or more ports of the switch are connected to the same LAN; the LAN where the bridge serves as a designated switch.
Alternate port	Serves as an alternate port for the root port providing a redundant path towards the root bridge.

Only the root port and the designated ports are part of the active topology; the alternate and backup ports do not participate in it. When the network is stable, the root and the designated ports are in the forwarding state, while the alternate and backup ports are in the discarding state. When there is a topology change, the new RSTP port roles allow a faster transition of an alternate port into the forwarding state.

For more information about spanning trees, see the introductory sections in the 802.1d Spanning Tree Protocol chapter.

RSTP parameters

The parameters you would normally set when you configure STP are applicable to RSTP. Before you configure RSTP see the STP parameters sections for descriptions of the bridge parameters. the error disable timeout parameter and the port channel path cost parameter.

There is one parameter that can be configured in RSTP that is not available in STP; the transmit hold count. This parameter configures the BPDU burst size by specifying the maximum number of BPDUs transmitted per second for before pausing for 1 second. The range is 1 through 10 while the default is 6. See the section Enabling RSTP and configuring RSTP parameters for the procedure to configure this parameter.

The edge port and auto edge features can be enabled in RSTP as well. See the section Edge port and automatic edge detection and the section Configuring RSTP on an interface for descriptions of these features and how they are configured.

Edge port and automatic edge detection

Configuring the edge port feature makes a port transition directly from initialization to the forwarding state, skipping the listening and learning states.

From an interface, you can configure a device to automatically identify the edge port. The port can become an edge port if no BPDU is received. By default, automatic edge detection is disabled.

Follow these guidelines to configure a port as an edge port:

- When edge port is enabled, the port still participates in a spanning tree.
- A port can become an edge port if no BPDU is received.
- When an edge port receives a BPDU, it becomes a normal spanning tree port and is no longer an edge port.
- Because ports that are directly connected to end stations cannot create bridging loops in the network, edge ports transition directly to the forwarding state and skip the listening and learning states.

  NOTE
  If BPDUs are received on a port fast enabled interface, the interface loses the edge port status unless it receives a **shutdown** or **no shutdown** command.

Configuring RSTP

Enabling and configuring RSTP globally

Follow these steps to enable and configure RSTP.

See the section STP parameters for parameters applicable to all STP variants.

You can enable RSTP or RSTP with one or more parameters enabled. The parameters can be enabled or changed individually by entering the commands in steps 1 and 2, running the parameter command, verifying the result, and then saving the configuration.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable RSTP.

```
device(config)# protocol spanning-tree rstp
```

   You can shut down RSTP by entering the **shutdown** command.

3. Designate the root device.

```
device(conf-rstp)# bridge-priority 28582
```

   The range is 0 through 61440 and the priority values can be set only in increments of 4096.

   You can shut down RSTP by entering the **shutdown** command when in RSTP configuration mode.

4. Configure the bridge forward delay value.

```
device(conf-rstp)# forward-delay 15
```

5. Configure the bridge maximum aging time value.

```
device(conf-rstp)# max-age 20
```

6. Enable the error disable timeout timer.
   a) Enable the timer.

```
device(conf-rstp)# error-disable-timeout enable
```

   b) Configure the error disable timeout interval value.

```
device(conf-rstp)# error-disable-timeout interval 60
```

7. Configure the port-channel path cost.

```
device(conf-rstp)# port-channel path-cost custom
```

8. Configure the bridge hello-time value.

```
device(conf-rstp)# hello-time 2
```

9. Specify the transmit hold count.

```
device(config-rstp)# transmit-holdcount 5
```

   This command configures the maximum number of BPDUs transmitted per second.

10. Return to privileged exec mode.

```
device(conf-rstp)# end
```

11. Verify the configuration

```
device# show spanning-tree

 Spanning-tree Mode: Rapid Spanning Tree Protocol

 Root Id: 8000.01e0.5200.0180 (self)
 Bridge Id: 8000.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: enabled
 Bpdu-guard errdisable timeout interval: 60 sec
```

    Observe that the settings comply with the formula set out in the STP parameters section, as:

    (2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

    or in this case: 28 ≥ 20 ≥ 6.

12. Save the configuration.

```
device# copy running-config startup-config
```

Enabling RSTP and configuring RSTP parameters example

```
device# configure terminal
device(config)# protocol spanning-tree rstp
device(conf-rstp)# bridge-priority 28582
device(conf-rstp)# forward-delay 20
device(conf-rstp)# max-age 25
device(conf-rstp)# error-disable-timeout enable
device(conf-rstp)# error-disable-timeout interval 60
device(conf-rstp)# port-channel path-cost custom
device(conf-rstp)# hello-time 5 forward-delay 16 max-age 21
device(conf-rstp)# transmit-holdcount 5
device(conf-rstp# end
device# show spanning-tree
device# copy running-config startup-config
```

Enabling and configuring RSTP on an interface

Follow these steps to configure RSTP on an Ethernet interface.

You can configure the parameters individually on an interface by doing the following:

1. Entering the commands in Steps 1 through 3.
2. Specifying additional parameters, as appropriate.
3. Verifying the result.
4. Saving the configuration.

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface subtype configuration mode.

```
device(config)# interface ethernet 0/10
```

3. Enable the interface.

```
device(conf-if-eth-0/10)# no shutdown
```

To disable the spanning tree on the interface you use the **spanning-tree shutdown** command.

4. Specify the port priority on the interface.

```
device(conf-if-eth-0/10)# spanning-tree priority 128
```

The range is from 0 through 240 in increments of 16. The default value is 128.

5. Specify the path cost on the interface.

```
device(conf-if-eth-0/10)# spanning-tree cost 20000000
```

The lower the path cost means a greater chance that the interface becomes the root port. The range is 1 through 200000000. The default path cost is assigned as per the port speed.

6. Enable edge port.

```
device(conf-if-eth-0/10)# spanning-tree edgeport
```

If BPDUs are received on a port fast enabled interface, the interface loses the edge port status unless it receives a **shutdown** or **no shutdown** command.

7. Enable BPDU guard on the interface.

```
device(conf-if-eth-0/10)# spanning-tree edgeport bpdu-guard
```

BPDU guard removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

8. Enable automatic edge detection on the interface.

```
device(conf-if-eth-0/10)# spanning-tree autoedge
```

You use this command to automatically identify the edge port. A port becomes an edge port if it receives no BPDUs. By default, automatic edge detection is disabled.

9. Enable root guard on the interface.

```
device(conf-if-eth-0/10)# spanning-tree guard root
```

Root guard protects the root bridge from malicious attacks and unintentional misconfigurations where a bridge device that is not intended to be the root bridge becomes the root bridge.

10. Specify a link type on the interface.

```
device(conf-if-eth-0/10)# spanning-tree link-type point-to-point
```

NOTE
The link type is explicitly configured as **point-to-point** rather than **shared**.

11. Return to privileged EXEC mode.

```
device(conf-if-eth-0/10)# end
```

12. Verify the configuration.

```
device# show spanning-tree

 Spanning-tree Mode: Rapid Spanning Tree Protocol

 Root Id: 8000.01e0.5200.0180 (self)
 Bridge Id: 8000.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: enabled
 Bpdu-guard errdisable timeout interval: 60 sec


 Port Eth 0/10 enabled
    Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Spanning Tree Protocol - Received None - Sent STP
    Edgeport: on; AutoEdge: yes; AdminEdge: no; EdgeDelay: 3 sec
    Configured Root guard: on; Operational Root guard: on
    Bpdu-guard: on
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

Interface   Role     Sts     Cost       Prio     Link-type     Edge
-------------------------------------------------------------------
Eth 0/10    DES      FWD     20000000   128      P2P           No
```

The **forward-delay**, **hello-time**, and **max-age** parameters are set globally, not on the interface.

Observe that the settings comply with the formula set out in the STP parameters section, as:

(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

or in this case: 28 ≥ 20 ≥ 6.

13. Save the configuration.

```
device# copy running-config startup-config
```

RSTP on an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 0/10
device(conf-if-eth-0/10)# no spanning-tree shutdown
device(conf-if-eth-0/10)# spanning-tree priority 128
device(conf-if-eth-0/10)# spanning-tree cost 20000000
device(conf-if-eth-0/10)# spanning-tree edgeport
device(conf-if-eth-0/10)# spanning-tree edgeport bpdu-guard
device(conf-if-eth-0/10)# spanning-tree autoedge
device(conf-if-eth-0/10)# spanning-tree guard root
device(conf-if-eth-0/10)# spanning-tree link-type point-to-point
device(conf-if-eth-0/10)# end
device# show spanning-tree
device# copy running-config startup-config
```

Configuring basic RSTP parameters

Follow these steps to configure basic RSTP parameters.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable RSTP.

```
device(config)# protocol spanning-tree rstp
```

3. Designate the root device.

```
device(conf-rstp)# bridge-priority 28582
```

4. Enable the error disable timeout timer value.

```
device(conf-rstp)# error-disable-timeout enable
```

5. Configure the error-disable-timeout interval value.

```
device(conf-rstp)# error-disable-timeout interval 60
```

6. Enable edge port on switch ports.
   a) Enter interface subtype configuration mode for the switchport.

```
device(conf-rstp)# interface ethernet 0/10
```

   b) Enable edge port.

```
device(conf-if-eth-0/10)# spanning-tree edge-port
```

   c) Return to global configuration mode.

```
device(conf-if-eth-0/10)# exit
```

   d) Repeat the above steps for all ports that connect to a workstation or PC.
7. Specify port priorities.
   a) Enter interface subtype configuration mode.

```
device(config)# interface ethernet 0/11
```

   b) Configure the port priority.

```
device(conf-if-eth-0/11)# spanning-tree priority 1
```

   c) Return to global configuration mode.

```
device(conf-if-eth-0/11)# exit
```

8. Enable the guard root feature.
   a) Enter interface configuration mode.

```
device(config)# interface ethernet 0/1
```

   b) Configure the port priority.

```
device(conf-if-eth-0/1)# spanning-tree guard root
```

   c) Return to privileged EXEC mode.

```
device(conf-if-eth-0/1)# exit
```

9. Verify the configuration.

```
device# show spanning-tree

Spanning-tree Mode: Rapid Spanning Tree Protocol

Root Id: 4096.01e0.5200.0180 (self)
Bridge Id: 4096.01e0.5200.0180
Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
Number of topology change(s): 0
Bpdu-guard errdisable timeout: disabled
Bpdu-guard errdisable timeout interval: 300 sec
switch# show spanning-tree brief
Spanning-tree Mode: Rapid Spanning Tree Protocol
Root ID Priority 4096
Address 768e.f805.5800
Hello Time 2, Max Age 20, Forward Delay 15
Bridge ID Priority 4096
Address 768e.f805.5800

Interface   Role     Sts     Cost     Prio     Link-type     Edge
-------------------------------------------------------------------
Eth 0/1     DES      FWD     2000     128      P2P           No
Eth 0/10    DES      FWD     2000     128      P2P           No
Eth 0/11    DES      FWD     2000     128      P2P           No
```

   Observe that the settings comply with the formula set out in the STP parameters section, as follows:

   (2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

   or in this case: 28 ≥ 20 ≥ 6.

10. Save the configuration.

```
device# copy running-config startup-config
```

Basic RSTP configuration example

```
device# configure terminal
device(config)# protocol spanning-tree rstp
device(conf-rstp)# bridge-priority 28582
device(conf-rstp)# error-disable-timeout enable
device(conf-rstp)# error-disable-timeout interval 60
device(conf-rstp)# interface etherne 0/10
device(conf-if-eth-0/10)# spanning-tree edge-port
device(conf-if-eth-0/10)# exit
device(config)# interface ethernet 0/11
device(conf-if-eth-0/11)# spanning-tree priority 1
device(conf-if-eth-0/11)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# spanning-tree guard root
device(conf-if-eth-0/1)# exit
device# show spanning-tree
device# copy running-config startup-config
```

Clearing spanning tree counters

Follow these steps to clear spanning tree counters on all interfaces or on the specified interface.

1. Clear spanning tree counters on all interfaces.

   ```
   device# clear spanning-tree counter
   ```

2. Clear spanning tree counters on a specified Ethernet interface.

   ```
   device# clear spanning-tree counter interface ethernet 0/3
   ```

3. Clear spanning tree counters on a specified port channel interface.

   ```
   device# clear spanning-tree counter interface port-channel 12
   ```

   Port channel interface numbers range from 1 through 64.

Clearing spanning tree-detected protocols

Follow these steps to restart the protocol migration process.

These commands force a spanning tree renegotiation with neighboring devices on either all interfaces or on a specified interface.

1. Restart the spanning tree migration process on all interfaces.

   ```
   device# clear spanning-tree detected-protocols
   ```

2. Restart the spanning tree migration process on a specific Ethernet interface.

   ```
   device# clear spanning-tree detected-protocols interface ethernet 0/3
   ```

3. Restart the spanning tree migration process on a specific port channel interface.

   ```
   device# clear spanning-tree detected-protocols port-channel 12
   ```

   Port channel interface numbers range from 1 through 64.

Shutting down RSTP

Follow these steps to shut down RSTP either globally or on a specific interface.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Shut down RSTP.
   - Shut down STP globally and return to privileged EXEC mode.

```
device(config)# protocol spanning-tree rstp
device(conf-rstp)# shutdown
device(conf-rstp)# end
```

   - Shut down RSTP on a specific interface and return to privileged EXEC mode.

```
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# spanning-tree shutdown
device(conf-if-eth-0/2)# end
```

   - Shut down RSTP on a specific VLAN and return to privileged EXEC mode.

```
device(config)# vlan 10
device(config-vlan-10)# spanning-tree shutdown
device(config-vlan-10)# end
```

3. Verify the configuration.

```
device# show spanning-tree
device#
```

4. Save the configuration.

```
device# copy running-config startup-config
```

Per-VLAN Spanning Tree+ and Rapid Per-VLAN Spanning Tree+

PVST+ and R-PVST+ overview

The Per-VLAN Spanning Tree Plus (PVST+) protocol runs a spanning tree instance for each VLAN in the network. The version of PVST+ that uses the RSTP state machine is called Rapid-PVST Plus (R-PVST+). R-PVST+ has one instance of spanning tree for each VLAN on the device.

Both the STP and the RSTP build a single logical topology. A typical network has multiple VLANs. A single logical topology does not efficiently utilize the availability of redundant paths for multiple VLANs. A single logical topology does not efficiently utilize the availability of redundant paths for multiple VLANs. If a port is set to the blocked state or the discarding state for one VLAN (under the STP or the RSTP), it is the same for all other VLANs. PVST+ builds on the STP on each VLAN, and R-PVST+ builds on the RSTP on each VLAN.

PVST+ R-PVST+ provide interoperability with Cisco PVST and R-PVST and other vendor switches which implement Cisco PVST or R-PVST. the PVST+ and R-PVST+ implementations are extensions to PVST and R-PVST, which can interoperate with an STP topology, including MSTP (CIST), on Extreme and other vendor devices sending untagged IEEE BPDUs.

PVST+ and R-PVST+ guidelines and restrictions

Consider the following when configuring PVST+ and R-PVST+:

- Extreme supports PVST+ and R-PVST+ only. The PVST and R-PVST protocols are proprietary to Cisco and are not supported.
- A port native VLAN is the native VLAN ID associated with a trunk port on an Extreme switch. This VLAN ID is associated with all untagged packets on the port. The default native VLAN ID for a trunk port is 1.
- IEEE compliant switches run just one instance of STP protocol shared by all VLANs, creating a Mono Spanning Tree (MST). A group of such switches running a single spanning tree forms an MST region.
- You can configure up to 128 PVST+ or R-PVST+ instances. If you have more than 128 VLANs configured on the switch and enable PVST then the first 128 VLANs are PVST/+ or R-PVST+ enabled.
- In PVST/+ or R-PVST+ mode, when you are connected to a Cisco or MLX switch, the Cisco proprietary MAC address to which the BPDUs are sent/processed must be explicitly configured on a per-port basis.
- In PVST/+ or R-PVST+ mode, when you connect to a Cisco switch using a trunk port, the Extreme switch must have a native VLAN configured on the trunk port (same configuration as on the other side).
- A Common Spanning Tree (CST) is the single spanning tree instance used by Extreme switches to interoperate with 802.1q bridges. This spanning tree instance stretches across the entire network domain (including PVST, PVST+ and 802.1q regions). It is associated with VLAN 1 on the Extreme switch.
- In order to interact with STP and IEEE 802.1q trunk, PVST evolved to PVST+ to interoperate with STP topology by STP BPDU on the native or default VLAN.
- A group of switches running PVST+ is called a PVST+ region.

For more information about spanning trees, see the introductory sections in the Spanning Tree Protocol chapter.

PVST+ and R-PVST+ parameters

The parameters you would normally set when you configure STP are applicable to PVST+ and R-PVST+. Before you configure PVST+ or R-PVST+ parameters see the sections in the Standing Tree Protocol chapter explaining bridge parameters, the error disable timeout parameter and the port channel path cost parameter.

There is one parameter that can be configured in R-PVST+ that is not available in STP or PVST+; the transmit hold count. This parameter configures the BPDU burst size by specifying the maximum number of BPDUs transmitted per second for before pausing for 1 second. The range is 1 through 10 while the default is 6. See the section Configuring R-PVST+ for the procedure to configure this parameter.

Bridge protocol data units in different VLANs

PVST+ uses the spanning tree instance for VLAN 1 to join the CST in the network to build the CST, PVST+ processes and sends standard IEEE Bridge protocol data units (BPDUs) on all the ports in VLAN 1 (access/trunk).

Across IEEE 802.1q trunks, Extreme switches run PVST+. The goal is to interoperate with standard IEEE STP (or RSTP or MSTP), while transparently tunneling PVST+ instance BPDUs across the MST region to potentially connect to other Extreme switches across the MST region.

On trunk ports that allow VLAN 1, PVST+ also sends PVST+ BPDUs to a Cisco-proprietary multicast MAC address (0100.0ccc.cccd) or Extreme-proprietary multicast MAC address (0304.0800.0700) depending on the configuration. By default, the PVST+ BPDUs are sent to Extreme-proprietary multicast MAC address on Extreme switches. These BPDUs are tunneled across an MST region. The PVST + BPDUs for VLAN 1 are only used for the purpose of consistency checks and that it is only the IEEE BPDUs that are used for building the VLAN 1 spanning tree. So in order to connect to the CST, it is necessary to allow VLAN 1 on all trunk ports.

For all other VLANs, PVST+ BPDUs are sent on a per-VLAN basis on the trunk ports. These BPDUs are tunneled across an MST region. Consequently, for all other VLANs, MST region appears as a logical hub. The spanning tree instances for each VLAN in one PVST+ region map directly to the corresponding instances in another PVST+ region and the spanning trees are calculated using the per-VLAN PVST+ BPDUs.

Similarly, when a PVST+ region connects to a MSTP region, from the point of view of MSTP region, the boundary bridge thinks it is connected to a standard IEEE compliant bridge sending STP BPDUs. So it joins the CIST of the MSTP region to the CST of the PVST+ region (corresponding to VLAN 1). The PVST+ BPDUs are tunneled transparently through the MSTP region. So from the Extreme bridge point of view, the MSTP region looks like a virtual hub for all VLANs except VLAN 1.

The PVST+ BPDUs are sent untagged for the native VLAN and tagged for all other VLANs on the trunk port.

On access ports, Extreme switches run classic version of IEEE STP/RSTP protocol, where the BPDUs are sent to the standard IEEE multicast address "0180.C200.0000". So if we connect a standard IEEE switch to an access port on the Extreme switch, the spanning tree instance (corresponding to the access VLAN on that port) of the Extreme switch is joined with the IEEE STP instance on the adjacent switch.

For introductory information about STP BPDUs, see the section BPDUs on page 200.

BPDU configuration notes

In order to build a spanning tree for the bridge topology, the bridges must exchange control frames. These frames are called Bridge Protocol data units (BPDU).

BPDUs are sent to a Cisco-proprietary multicast MAC address 0100.0ccc.cccd or Extreme-proprietary multicast MAC address 0304.0800.0700.By default, the PVST+ BPDUs are sent to Extreme-proprietary multicast MAC address on Extreme switches. These are called SSTP (Single Spanning Tree Protocol) BPDUs. The format of the SSTP BPDU is nearly identical to the 802.1d BPDU after the SNAP header, except that a type-length-value (TLV) field is added at the end of the BPDU. The TLV has 2 bytes for type (0x0), 2 bytes for length, and 2 bytes for the VLAN ID. See Extreme BPDU PVST+ headers/fields on page 227 and BPDU R-PVST+ header and field comparisons on page 227 for an outline of the BPDU header content.

Topology Change Notification (TCN) BPDUs are used to inform other switches of port changes. TCNs are injected into the network by a non-root switch and propagated to the root. Upon receipt of the TCN, the root switch will set a Topology Change flag in its normal BPDUs. This flag is propagated to all other switches to instruct them to rapidly age out their forwarding table entries.

In PVST+, three types of TCN BPDUs are sent out depending on the type of the link. See Extreme PVST+ TCN BPDU headers/fields on page 229 and Cisco PVST TCN BPDU headers/fields on page 230.

- Standard IEEE TCN BPDU.
- Untagged TCN BPDU sent to the Cisco/Extreme proprietary MAC address.
- Tagged TCN BPDU sent to the Cisco/Extreme proprietary MAC address.

BPDU R-PVST+ header and field comparisons

These tables outline the differences between Extreme R-PVST+ BPDU and Cisco R-PVST+ BPDU header fields.

Extreme R-PVST+ BPDU headers/fields

Header/field	Standard IEEE STP/RSTP BPDU (64B padded)	R-PVST+ untagged BPDU (64B padded)	R-PVST+ tagged BPDU (72B padded)
Source Address (MAC SA)	6B	6B	6B
Destination Address (MAC DA)	0180C2.000000 (6B)	030408.000700 (6B)	030408.000700 (6B)
Length	2B	2B	-
Type	-	-	81 00 (2B)
802.1q tag	-	-	4B
Source Service Access Point (SSAP)	42	AA 03	AA 03
Destination Service Access Point (DSAP)	42	AA	AA
Extreme Organizationally Unique Identifier (OUI)	-	02 04 08	02 04 08
PVST PID	-	01 0B	01 0B
Logical Link Control (LLC)	3B	+	+
SubNetwork Access Protocol (SNAP)	-	Yes (2B)	Yes (2B)
IEEE BPDU INFO	35B	35B	35B
Type, Length, Value (TLV) Pad Type Length	-	6B 00 (1B) 00 00 00 02	6B 00 (1B) 00 00 00 02

Header/field	Standard IEEE STP/RSTP BPDU (64B padded)	R-PVST+ untagged BPDU (64B padded)	R-PVST+ tagged BPDU (72B padded)
VLAN ID		2B	2B

Cisco R-PVST+ BPDU headers/fields

Header/field	Standard IEEE STP/RSTP BPDU (64B padded)	R-PVST+ untagged BPDU (64B padded)	R-PVST+ tagged BPDU (72B padded)
MAC SA	6B	6B	6B
MAC DA	0180C2.000000 (6B)	01000C.CCCCCD (6B)	010002.CCCCCD (6B)
Length	2B	2B	-
Type	-	-	81 00 (2B)
802.1q tag	-	-	4B
SSAP	42 03	AA 03	AA 03
DSAP	42	AA	AA
Cisco OUI	-	00 00 0C	00 00 0C
PVST PID	-	01 0B	01 0B
LLC	3B	+	+
SNAP	-	Yes	Yes
IEEE BPDU INFO	35B	35B	35B
TLV Pad Type Length VLAN ID	-	6B 00 (1B) 00 00 00 02 2B	6B 00 (1B) 00 00 00 02 2B

Sent BPDUs

On an 802.1q trunk, the PVST+ enabled switch sends the following BPDUs:

1. For all tagged VLANs on the port on which PVST+ is enabled, 802.1q tagged SSTP BPDUs are sent to the Cisco or Extreme MAC address. The 802.1q tag contains the VLAN ID. (VLAN 1 could be tagged on the port. In that case a tagged BPDU for VLAN 1 is sent). The IEEE compliant switches do not consider these BPDUs as a control packet. So they forward the frame as they would forward to any unknown multicast address on the specific VLAN.
2. If PVST+ is enabled on the untagged (native) VLAN of the port, an untagged SSTP BPDU is sent to the Extreme or Cisco MAC address on the native VLAN of the trunk. It is possible that the native VLAN on the Extreme or Cisco port is not VLAN 1. This BPDU is also forwarded on the native VLAN of the IEEE 802.1q switch just like any other frame sent to an unknown multicast address.
3. In addition to the above SSTP BPDUs, a standard IEEE BPDU (802.1d) is also sent, corresponding to the information of VLAN 1 on the Extreme or Cisco switch. This BPDU is not sent if VLAN 1 is explicitly disabled on the trunk port.

The following table lists the types of BPDUs sent in case of different port types. The numbers in the third column are the VLAN instance for which these BPDUs are sent/processed.

TABLE 28 Types of BPDUs sent for different port types

Port Configuration	Extreme or Cisco - PVST(+)	VLAN instance
Access - VLAN 1	Standard IEEE BPDU (64B)	1
Access - VLAN 100	Standard IEEE BPDU (64B)	100
Trunk - Native VLAN 1 Allowed VLANs - 1, 100, 200	Standard IEEE BPDU (64B) Extreme or Cisco untagged BPDU (68B) Extreme or Cisco tagged BPDU (72B) Extreme or Cisco tagged BPDU (72B)	1 1 100 200
Trunk - Native VLAN 100 Allowed VLANs - 1, 100, 200	Standard IEEE BPDU (64B) Extreme or Cisco untagged BPDU (68B) Extreme or Cisco tagged BPDU (72B Extreme or Cisco tagged BPDU (72B)	1 100 1 200
Trunk - Native VLAN 100 Allowed VLANs - 100	Extreme or Cisco untagged BPDU (68B)	100
Trunk - Native VLAN 100 Allowed VLANs - 100, 200	Extreme or Cisco untagged BPDU (68B) Extreme or Cisco tagged BPDU (72B)	100 200

TCN headers and fields

Since PVST+ is based on STP, and Rapid-PVST+ is based on RSTP, TCN BPDUs are sent only in PVST+ and not in Rapid-PVST+ mode.

For introductory information about STP BPDUs, see the section TCN BPDUs on page 201.

Extreme PVST+ TCN BPDU headers/fields

Header/field	Standard IEEE STP TCN BPDU (64B with padding)	PVST+ untagged TCN BPDU (64B with padding)	PVST+ tagged TCN BPDU (68B with padding)
MAC SA	6B	6B	6B
MAC DA	0180C2.000000 (6B)	030408.000700 (6B)	030408.000700 ((6B)
Length	2B	2B	-
Type	-	-	81 00 (2B)
802.1q tag	-	-	4B
SSAP	42 03	AA 03	AA 03
DSAP	42	AA	AA
Cisco OUI	-	02 04 08	02 04 08
PVST PID	-	01 0B	01 0B
LLC	3B	8B	8B
SNAP	4B	Entire BPDU with type = TCN 35B	Entire BPDU with type = TCN 35B

Cisco PVST TCN BPDU headers/fields

Header/field	Standard IEEE STP TCN BPDU (64B padded)	PVST untagged TCN BPDU (64B padded)	PVST tagged TCN BPDU (68B padded)
MAC SA	6B	6B	6B
MAC DA	0180C2.000000 (6B)	01000C.CCCCCD (6B)	01000C.CCCCCD (6B)
Length	2B	2B	-
Type	-	-	81 00 (2B)
802.1q tag	-	-	4B
SSAP	42 03	AA 03	AA 03
DSAP	42	AA	AA
Cisco OUI	-	00 00 0C	00 00 0C
PVST PID	-	01 0B	01 0B
LLC	3B	8B	8B
SNAP	-	Yes	Yes
IEEE TCN BPDU INFO	4B	Entire BPDU with type = TCN 35B	Entire BPDU with type = TCN 35B

PortFast

PortFast allows an interface to transition quickly to the forwarding state.

Consider the following when configuring PortFast:

- Do not enable PortFast on ports that connect to other devices.
- PortFast only needs to be enabled on ports that connect to workstations or PCs. Repeat this configuration for every port connected to workstations or PCs.
- Enabling PortFast on ports can cause temporary bridging loops, in both trunking and nontrunking mode.
- If BPDUs are received on a PortFast- enabled interface, the interface loses the edge port status unless it receives a **shutdown**/**no shutdown** command.
- PortFast immediately puts the interface into the forwarding state without having to wait for the standard forward time.

Edge port and automatic edge detection

Configuring the edge port feature makes a port transition directly from initialization to the forwarding state, skipping the listening and learning states.

From an interface, you can configure a device to automatically identify the edge port. The port can become an edge port if no BPDU is received. By default, automatic edge detection is disabled.

Follow these guidelines to configure a port as an edge port:

- When edge port is enabled, the port still participates in a spanning tree.
- A port can become an edge port if no BPDU is received.
- When an edge port receives a BPDU, it becomes a normal spanning tree port and is no longer an edge port.
- Because ports that are directly connected to end stations cannot create bridging loops in the network, edge ports transition directly to the forwarding state and skip the listening and learning states.

NOTE
If BPDUs are received on a port fast enabled interface, the interface loses the edge port status unless it receives a **shutdown** or **no shutdown** command.

Configuring PVST+ and R-PVST+

Enabling and configuring PVST+ globally

Use this procedure to enable and set parameters for PVST+.

You can enable PVST+ with one or more parameters configured. The parameters can be configured or changed individually by entering the commands in steps 1 and 2, running the parameter command, verifying the result, and then saving the configuration.

For more information about spanning trees and spanning tree parameters, see the introductory sections in the Spanning Tree Protocol chapter.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable PVST+.

```
device(config)# protocol spanning-tree pvst
```

3. Configure the bridge priority for the common instance.

```
device(config-pvst)# bridge-priority 4096
```

   Valid values range from 0 through 61440 in increments of 4096. Assigning a lower priority value indicates that the bridge might become root.

   You can shut down PVST+ by entering the **shutdown** command when in PVST configuration mode.

4. Configure the forward delay parameter.

```
device(config-pvst)# forward-delay  11
```

5. Configure the hello time parameter.

```
device(config-pvst)# hello-time 2
```

6. Configure the maximum age parameter.

```
device(config-pvst)# max-age 7
```

7. Return to privileged exec mode.

```
device(config-pvst)# end
```

8. Verify the configuration.

```
device#  show spanning-tree brief
VLAN 1

 Spanning-tree Mode: PVST Protocol

      Root ID      Priority 4097
                   Address 01e0.5200.0180
                   Hello Time 2, Max Age 7, Forward Delay 11

      Bridge ID    Priority 4097
                   Address 01e0.5200.0180
                   Hello Time 2, Max Age 7, Forward Delay 11

 Interface     Role  Sts  Cost       Prio  Link-type        Edge
 ------------------------------------------------------------------

VLAN 100

 Spanning-tree Mode: PVST Protocol

      Root ID      Priority 4196
                   Address 01e0.5200.0180
                   Hello Time 2, Max Age 7, Forward Delay 11

      Bridge ID    Priority 4196
                   Address 01e0.5200.0180
                   Hello Time 2, Max Age 7, Forward Delay 11

 Interface     Role  Sts  Cost       Prio  Link-type        Edge
 ------------------------------------------------------------------
```

   Observe that the settings comply with the formula set out in the STP parameters section, as:

   (2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

   or in this case: 20 ≥7 ≥ 6.

9. Save the configuration.

```
device# copy running-config startup-config
```

PVST+ configuration example

```
device# configure terminal
device(config)# protocol spanning-tree pvst
device(config-pvst)# bridge-priority 4096
device(config-pvst)# forward-delay 11
device(config-pvst)# hello-time 2
device(config-pvst)# max-age 7
device(config-pvst)# end
device# show spanning-tree brief
device# copy running-config startup-config
```

For more information about configuring PVST+ parameters, see STP parameters on page 204. PVST+, R-PVST+, and other types of spanning trees share many tasks with STP.

Enabling and configuring PVST+ on an interface

Follow these steps to enable and configure PVST+ on an interface.

The ports and parameters can be configured individually on a system by:

1. Entering the commands in steps 1, and 2

2. Running the relevant addition steps and parameter commands
3. Verifying the result
4. Saving the configuration

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable PVST+.

```
device(config)# protocol spanning-tree pvst
```

3. Enter interface configuration mode.

```
device(config-pvst)# interface ethernet 0/3
```

4. Enable spanning tree on the interface.

```
device(conf-if-eth-0/3)# no spanning-tree shutdown
```

5. Configure the interface link type.

```
device(conf-if-eth-0/3)# spanning-tree link-type point-to-point
```

6. Specify the port priority to influence the selection of root or designated ports.

```
device(conf-if-eth-0/3)# spanning-tree priority 64
```

   The range is from 0 through 240 in increments of 16. The default value is 128.

7. Configure the path cost for spanning tree calculations on the interface.

```
device(conf-if-eth-0/3)# spanning-tree cost 10000
```

   The lower the path cost means a greater chance that the interface becomes the root port. The range is 1 through 200000000. The default path cost is assigned as per the port speed.

8. Configure the path cost for spanning tree calculations a specific VLAN.

```
device(conf-if-eth-0/3)# spanning-tree vlan 10 cost 10000
```

   The lower the path cost means a greater chance that the interface becomes the root port. The range is 1 through 200000000. The default path cost is assigned as per the port speed.

9. Enable root guard on the interface.

```
device(conf-if-eth-0/3)# spanning-tree guard root
```

   Root guard protects the root bridge from malicious attacks and unintentional misconfigurations where a bridge device that is not intended to be the root bridge becomes the root bridge.

10. Enable BPDU guard on the interface.

```
device(conf-if-eth-0/3)# spanning-tree port-fast bpdu-guard
```

    BPDU guard removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

11. Enable BPDU filtering on the interface.

```
device(conf-if-eth-0/3)# spanning-tree port-fast bpdu-filter
```

    BPDU filtering allows you to avoid transmitting BPDUs on ports that are connected to an end system.

12. Return to privileged EXEC mode.

```
device(conf-if-eth-0/3)# exit
```

13. Verify the configuration.

```
device# show spanning-tree brief

 Spanning-tree Mode: PVST Protocol

      Root ID      Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

      Bridge ID    Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

 Interface    Role  Sts  Cost       Prio  Link-type     Edge
 -------------------------------------------------------------
 Eth 0/3      DES   FWD  200000     64    P2P            No
```

    Observe that the settings comply with the formula set out in the STP parameters section, as:

    (2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

    or in this case :38 ≥ 25 ≥ 18.

14. Save the configuration.

```
device# copy running-config startup-config
```

PVST+ on an interface configuration example

```
device# configure terminal
device(config)# protocol spanning-tree pvst
device(conf-pvst)# interface ethernet 0/3
device(conf-if-eth-0/3)# no spanning-tree shutdown
device(conf-if-eth-0/3)# spanning-tree link-type point-to-point
device(conf-if-eth-0/3)# spanning-tree priority 64
device(conf-if-eth-0/3)# spanning-tree cost 10000
device(conf-if-eth-0/3)# spanning-tree vlan 10 cost 10000
device(conf-if-eth-0/3)# spanning-tree guard root
device(conf-if-eth-0/3)# spanning-tree port-fast bpdu-guard
device(conf-if-eth-0/3)# exit
device# show spanning-tree
device# copy running-config startup-config
```

Enabling and configuring PVST+ on a system

Follow the steps to configure PVST+ on a system.

The ports and parameters can be configured individually on a system by:

1. Entering the commands in steps 1, and 2
2. Running the relevant addition steps and parameter commands

3. Verifying the result
4. Saving the configuration

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable PVST+.

```
device(config)# protocol spanning-tree pvst
```

3. Configure the bridge priority for the common instance.

```
device(config-pvst)# bridge-priority 4096
```

   Valid values range from 0 through 61440 in multiples of 4096. Assigning a lower priority value indicates that the bridge might become root.

4. Configure the forward delay parameter.

```
device(config-pvst)# forward-delay  15
```

5. Configure the hello time parameter.

```
device(config-pvst)# hello-time 2
```

6. Configure the maximum age parameter.

```
device(config-pvst)# max-age 20
```

7. Add VLANs.
   a) Configure VLAN 100 with a priority of 0.

```
device(config-pvst)# vlan 100 priority 0
```

      The bridge priority in configured in multiples of 4096.

   b) Configure VLAN 201 with a priority of 12288.

```
device(config-pvst)# vlan 201 priority 12288
```

   c) Configure VLAN 301 with a priority of 20480.

```
device(config-pvst)# vlan 301 priority 20480
```

8. Set the switching characteristics for interface 0/3.
   a) Enter interface configuration mode.

```
device(config)# interface ethernet 0/3
```

   b) Set the switching characteristics of the interface.

```
device(conf-if-eth-0/3)# switchport
```

   c) Set the interface mode to trunk.

```
device(conf-if-eth-0/3)# switchport mode trunk
```

   d) Add VLAN 100 as a member VLAN.

```
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 100
```

   e) Add VLAN 201 as a member VLAN.

```
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 201
```

   f) Add VLAN 301 as a member VLAN.

```
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 301
```

   g) Enable spanning tree on the interface.

```
device(conf-if-eth-0/3)# no spanning-tree shutdown
```

   h) Return to privileged EXEC mode.

```
device(conf-if-eth-0/3)# exit
```

9. Set the switching characteristics for interface 0/4.
   a) Enter interface configuration mode.

```
device(config)# interface ethernet 0/4
```

   b) Set the switching characteristics of the interface.

```
device(conf-if-eth-0/4)# switchport
```

   c) Set the interface mode to trunk.

```
device(conf-if-eth-0/4)# switchport mode trunk
```

   d) Add VLAN 100 as a member VLAN.

```
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 100
```

   e) Add VLAN 201 as a member VLAN.

```
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 201
```

   f) Add VLAN 301 as a member VLAN.

```
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 301
```

   g) Enable spanning tree on the interface.

```
device(conf-if-eth-0/4)# no spanning-tree shutdown
```

   h) Return to privileged EXEC mode.

```
device(conf-if-eth-0/4)# exit
```

10. To interoperate with switches other than VDX switches in PVST+ mode, you must configure the interface that is connected to that switch.
   a) Enter interface configuration mode for the port that interoperates with a VDX device.

```
device(config)# interface ethernet 0/12
```

   b) Specify the MAC address for the device.

```
device(conf-if-eth-0/12)# spanning-tree bpdu-mac 0100.0ccc.cccd
```

   c) Enable spanning tree on the interface.

```
device(conf-if-eth-0/12)# no spanning-tree shutdown
```

   d) Return to privileged EXEC mode.

```
device(conf-if-eth-0/12)# end
```

11. Verify the configuration.

```
device# show spanning-tree

VLAN 1

 Spanning-tree Mode: PVST Protocol

 Root Id: 0001.01e0.5200.0180 (self)
 Bridge Id: 0001.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec

 Port Et 0/3 enabled
    Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

 Port Et 0/4 enabled
    Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

VLAN 100

 Spanning-tree Mode: PVST Protocol

 Root Id: 0064.01e0.5200.0180 (self)
 Bridge Id: 0064.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec

 Port Et 0/3 enabled
    Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
```

```
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

 Port Et 0/4 enabled
    Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

VLAN 201

 Spanning-tree Mode: PVST Protocol

 Root Id: 30c9.01e0.5200.0180 (self)
 Bridge Id: 30c9.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec

 Port Et 0/3 enabled
    Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

 Port Et 0/4 enabled
    Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

VLAN 301

 Spanning-tree Mode: PVST Protocol

 Root Id: 512d.01e0.5200.0180 (self)
 Bridge Id: 512d.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec
```

```
Port Et 0/3 enabled
   Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
   Designated Path Cost: 0
   Configured Path Cost: 20000000
   Designated Port Id: 0; Port Priority: 128
   Designated Bridge: 0000.0000.0000.0000
   Number of forward-transitions: 0
   Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
   Portfast: off
   Configured Root guard: off; Operational Root guard: off
   Bpdu-guard: off
   Link-type: point-to-point
   Received BPDUs: 0; Sent BPDUs: 0

Port Et 0/4 enabled
   Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
   Designated Path Cost: 0
   Configured Path Cost: 20000000
   Designated Port Id: 0; Port Priority: 128
   Designated Bridge: 0000.0000.0000.0000
   Number of forward-transitions: 0
   Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
   Portfast: off
   Configured Root guard: off; Operational Root guard: off
   Bpdu-guard: off
   Link-type: point-to-point
   Received BPDUs: 0; Sent BPDUs: 0
```

Observe that the settings comply with the formula set out in the STP parameters section, as:

(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

or in this case: 28 ≥ 20 ≥ 6.

12. Save the configuration.

```
device# copy running-config startup-config
```

Enable PVST+ on a system configuration example

```
device# configure terminal
device(config)# protocol spanning-tree pvst
device(config-pvst)# bridge-priority 4096
device(config-pvst)# forward-delay  15
device(config-pvst)# hello-time 2
device(config-pvst)# max-age 20
device(config-pvst)# vlan 100 priority 0
device(config-pvst)# vlan 201 priority 12288
device(config-pvst)# vlan 301 priority 20480
device(config-pvst)# interface ethernet 0/3
device(conf-if-eth-0/3)# switchport
device(conf-if-eth-0/3)# switchport mode trunk
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 100
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 201
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 301
device(conf-if-eth-0/3)# no spanning-tree shutdown
device(conf-if-eth-0/3)# exit
device(config)# interface ethernet 0/4
device(conf-if-eth-0/4)# switchport
device(conf-if-eth-0/4)# switchport mode trunk
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 100
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 201
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 301
device(conf-if-eth-0/4)# no spanning-tree shutdown
device(conf-if-eth-0/4)# end
device# show spanning-tree
device# copy running-config startup-config
```

Enabling and configuring R-PVST+ globally

Use this procedure to enable the Rapid Per-VLAN Spanning Tree Protocol Plus (R-PVST+).

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable R-PVST+.

```
device(config)# protocol spanning-tree rpvst
```

3. Configure the bridge priority for the common instance.

```
device(config-rpvst)# bridge-priority 4096
```

   Valid priority values range from 0 through 61440 in multiples of 4096. Assigning a lower priority value indicates that the bridge might become root.

4. Configure the forward delay parameter.

```
device(config-rpvst)# forward-delay  20
```

5. Configure the hello time parameter.

```
device(config-rpvst)# hello-time 22
```

6. Configure the maximum age parameter.

```
device(config-rpvst)# max-age 8
```

7. Set the transmit hold count for the bridge.

```
device(config-rpvst)# transmit-holdcount 9
```

   This command configures the maximum number of BPDUs transmitted per second before pausing for 1 second. The range is 1 through 10. The default is 6.

8. Return to privileged exec mode.

```
device(config-rpvst)# end
```

9. Verify the configuration.

```
device# show spanning-tree brief
VLAN 1

 Spanning-tree Mode: Rapid PVST Protocol

      Root ID      Priority 4096
                   Address 01e0.5200.0180
                   Hello Time 2, Max Age 7, Forward Delay 11

      Bridge ID    Priority 32769
                   Address 01e0.5200.0180
                   Hello Time 8, Max Age 22, Forward Delay 20, Tx-HoldCount 9
                   Migrate Time 3 sec

 Interface     Role  Sts  Cost       Prio  Link-type        Edge
 ---------------------------------------------------------------------
```

   Observe that the settings comply with the formula set out in the STP parameters section, as:

   (2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

   or in this case: 20 ≥ 7 ≥ 6.

10. Save the configuration.

```
device# copy running-config startup-config
```

R-PVST+ configuration example

```
device# configure terminal
device(config)# protocol spanning-tree rpvst
device(config-rpvst)# bridge-priority 4096
device(config-rpvst)# forward-delay 20
device(config-rpvst)# hello-time 22
device(config-rpvst)# max-age 8
device(config-rpvst)# transmit-holdcount 9
device(config-rpvst)# end
device# show spanning-tree brief
device# copy running-config startup-config
```

For more information about configuring parameters, see the section STP parameter configuration.

Enabling and configuring R-PVST+ on an interface

Follow these steps to enable and configure R-PVST+ on an interface.

The ports and parameters can be configured individually on a system by:

1. Entering the commands in steps 1-3
2. Running the relevant addition steps and parameter commands
3. Verifying the result
4. Saving the configuration

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable R-PVST+.

```
device(config)# protocol spanning-tree rpvst
```

3. Enter interface configuration mode.

```
device(config-rpvst)# interface ethernet 0/3
```

4. Enable the spanning tree on the interface.

```
device(conf-if-eth-0/3)# no spanning-tree shutdown
```

5. Configure the interface link type.

```
device(conf-if-eth-0/3)# spanning-tree link-type point-to-point
```

6. Specify the port priority to influence the selection of root or designated ports.

```
device(conf-if-eth-0/3)# spanning-tree priority 64
```

   The range of priority values is from 0 through 240 in multiples of 16. The default value is 128.

7. Configure the path cost for spanning tree calculations on the interface.

```
device(conf-if-eth-0/3)# spanning-tree cost 200000
```

   The lower the path cost means a greater chance that the interface becomes the root port. The range is 1 through 200000000. The default path cost is assigned as per the port speed.

8. Configure the path cost for spanning tree calculations a specific VLAN.

```
device(conf-if-eth-0/3)# spanning-tree vlan 10 cost 10000
```

   The lower the path cost means a greater chance that the interface becomes the root port. The range is 1 through 200000000. The default path cost is assigned as per the port speed.

9. Enable automatic edge detection on the interface.

```
device(conf-if-eth-0/3)# spanning-tree autoedge
```

   You use this command to automatically identify the edge port. A port becomes an edge port if it receives no BPDUs. By default, automatic edge detection is disabled.

10. Enable root guard on the interface.

```
device(conf-if-eth-0/3)# spanning-tree guard root
```

    Root guard protects the root bridge from malicious attacks and unintentional misconfigurations where a bridge device that is not intended to be the root bridge becomes the root bridge.

11. Enable the spanning tree on the edge port.

```
device(conf-if-eth-0/3)# spanning-tree edgeport
```

    If BPDUs are received on a port fast enabled interface, the interface loses the edge port status unless it receives a **shutdown** or **no shutdown** command.

12. Enable BPDU guard on the interface.

```
device(conf-if-eth-0/3)# spanning-tree edgeport bpdu-guard
```

    BPDU guard removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

13. Return to privileged EXEC mode.

```
device(conf-if-eth-0/3)# exit
```

14. Verify the configuration.

```
device# show spanning-tree brief

 Spanning-tree Mode: Rapid PVST Protocol

      Root ID      Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

      Bridge ID    Priority 4096
                   Address 768e.f805.5800
                   Hello Time 8, Max Age 25, Forward Delay 20

 Interface    Role  Sts  Cost       Prio  Link-type     Edge
 ------------------------------------------------------------
 Eth 0/3      DES   FWD  200000     128   P2P            No
```

    Observe that the settings comply with the formula set out in the STP parameters section, as:

    (2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

    or in this case: 38 ≥ 25 ≥ 18.

15. Save the configuration.

```
device# copy running-config startup-config
```

R-PVST+ on an interface configuration example

```
device# configure terminal
device(config)# protocol spanning-tree rpvst
device(config-rpvst)# interface ethernet 0/3
device(conf-if-eth-0/3)# no spanning-tree shutdown
device(conf-if-eth-0/3)# spanning-tree link-type point-to-point
device(conf-if-eth-0/3)# spanning-tree priority 64
device(conf-if-eth-0/3)# spanning-tree cost 200000
device(conf-if-eth-0/3)# spanning-tree vlan 10 cost 10000
device(conf-if-eth-0/3)# spanning-tree autoedge
device(conf-if-eth-0/3)# spanning-tree guard root
device(conf-if-eth-0/3)# spanning-tree edgeport
device(conf-if-eth-0/3)# spanning-tree edgeport bpdu-guard
device(conf-if-eth-0/3)# exit
device# show spanning-tree
device# copy running-config startup-config
```

Enabling and configuring R-PVST+ on a system

Follow the steps to configure R-PVST+ on a system.

The ports and parameters can be configured individually by:

1. Entering the commands in steps 1 and 2
2. Running the relevant addition steps and parameter commands
3. Verifying the result
4. Saving the configuration

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable R-PVST+.

```
device(config)# protocol spanning-tree rpvst
```

   You can shut down R-PVST+ by entering the **shutdown** command when in `rpvst` configuration mode.

3. Configure the bridge priority for the common instance.

```
device(config-rpvst)# bridge-priority 4096
```

   Valid values range from 0 through 61440 in increments of 4096. Assigning a lower priority value indicates that the bridge might become root.

4. Configure the forward delay parameter.

```
device(config-rpvst)# forward-delay  20
```

5. Configure the hello time parameter.

```
device(config-rpvst)# hello-time 8
```

6. Configure the maximum age parameter.

```
device(config-rpvst)# max-age 22
```

7. Specify the transmit hold count.

```
device(config-rpvst)# transmit-holdcount 5
```

   This command configures the maximum number of BPDUs transmitted per second. The range of values is 1 through 10.

8. Configure VLANs.
   a) Configure VLAN 100 with a priority of 0.

```
device(config-rpvst)# vlan 100 priority 0
```

      Valid priority values range from 0 through 61440 in multiples of 4096.

   b) Configure VLAN 201 with a priority of 12288.

```
device(config-rpvst)# vlan 201 priority 12288
```

   c) Configure VLAN 301 with a priority of 20480.

```
device(config-rpvst)# vlan 301 priority 20480
```

9. Set the switching characteristics for interface 0/3.
   a) Enter interface configuration mode.

```
device(config-rpvst)# interface ethernet 0/3
```

   b) Set the switching characteristics of the interface.

```
device(conf-if-eth-0/3)# switchport
```

   c) Set the interface mode to trunk.

```
device(conf-if-eth-0/3)# switchport mode trunk
```

   d) Add VLAN 100 as a member VLAN.

```
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 100
```

   e) Add VLAN 201 as a member VLAN.

```
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 201
```

   f) Add VLAN 301 as a member VLAN.

```
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 301
```

   g) Enable spanning tree on the interface.

```
device(conf-if-eth-0/3)# no spanning-tree shutdown
```

   h) Return to privileged EXEC mode.

```
device(conf-if-eth-0/3)# exit
```

10. Set the switching characteristics for interface 0/4.
    a) Enter interface configuration mode.

```
device(config-rpvst)# interface ethernet 0/4
```

    b) Set the switching characteristics of the interface.

```
device(conf-if-eth-0/4)# switchport
```

    c) Set the interface mode to trunk.

```
device(conf-if-eth-0/4)# switchport mode trunk
```

    d) Add VLAN 100 as a member VLAN.

```
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 100
```

    e) Add VLAN 201 as a member VLAN.

```
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 201
```

    f) Add VLAN 301 as a member VLAN.

```
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 301
```

    g) Enable spanning tree on the interface.

```
device(conf-if-eth-0/4)# no spanning-tree shutdown
```

    h) Return to privileged EXEC mode.

```
device(conf-if-eth-0/4)# exit
```

11. To interoperate with switches other than VDX switches in R-PVST+ mode, you must configure the interface that is connected to that switch.
    a) Enter interface configuration mode for the port that interoperates with a VDX switch.

```
device(config)# interface ethernet 0/12
```

    b) Specify the MAC address for the device.

```
device(conf-if-eth-0/12)# spanning-tree bpdu-mac 0100.0ccc.cccd
```

    c) Enable spanning tree on the interface.

```
device(conf-if-eth-0/12)# no spanning-tree shutdown
```

    d) Return to privileged EXEC mode.

```
device(conf-if-eth-0/12)# end
```

12. Verify the configuration.

```
device# show spanning-tree

VLAN 1

 Spanning-tree Mode: Rapid PVST Protocol

 Root Id: 0001.01e0.5200.0180 (self)
 Bridge Id: 0001.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 20; Hello Time: 8; Max Age: 22; Max-hops: 20
 Tx-HoldCount 5
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec

 Port Et 0/3 enabled
    Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

 Port Et 0/4 enabled
    Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

VLAN 100

 Spanning-tree Mode: Rapid PVST Protocol

 Root Id: 0064.01e0.5200.0180 (self)
 Bridge Id: 0064.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 20; Hello Time: 8; Max Age: 22; Max-hops: 20
 Tx-HoldCount 5
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec

 Port Et 0/3 enabled
    Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
```

```
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

 Port Et 0/4 enabled
    Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

VLAN 201

 Spanning-tree Mode: Rapid PVST Protocol

 Root Id: 30c9.01e0.5200.0180 (self)
 Bridge Id: 30c9.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 20; Hello Time: 8; Max Age: 22; Max-hops: 20
 Tx-HoldCount 5
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec

 Port Et 0/3 enabled
    Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

 Port Et 0/4 enabled
    Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
    Designated Path Cost: 0
    Configured Path Cost: 20000000
    Designated Port Id: 0; Port Priority: 128
    Designated Bridge: 0000.0000.0000.0000
    Number of forward-transitions: 0
    Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
    Portfast: off
    Configured Root guard: off; Operational Root guard: off
    Bpdu-guard: off
    Link-type: point-to-point
    Received BPDUs: 0; Sent BPDUs: 0

VLAN 301

 Spanning-tree Mode: Rapid PVST Protocol

 Root Id: 512d.01e0.5200.0180 (self)
 Bridge Id: 512d.01e0.5200.0180

 Root Bridge Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 20; Hello Time: 8; Max Age: 22; Max-hops: 20
 Tx-HoldCount 5
```

```
Number of topology change(s): 0

Bpdu-guard errdisable timeout: disabled
Bpdu-guard errdisable timeout interval: 300 sec

Port Et 0/3 enabled
   Ifindex: 201351168; Id: 8001; Role: Disabled; State: Disabled
   Designated Path Cost: 0
   Configured Path Cost: 20000000
   Designated Port Id: 0; Port Priority: 128
   Designated Bridge: 0000.0000.0000.0000
   Number of forward-transitions: 0
   Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
   Portfast: off
   Configured Root guard: off; Operational Root guard: off
   Bpdu-guard: off
   Link-type: point-to-point
   Received BPDUs: 0; Sent BPDUs: 0

Port Et 0/4 enabled
   Ifindex: 201359360; Id: 8002; Role: Disabled; State: Disabled
   Designated Path Cost: 0
   Configured Path Cost: 20000000
   Designated Port Id: 0; Port Priority: 128
   Designated Bridge: 0000.0000.0000.0000
   Number of forward-transitions: 0
   Version: Per-VLAN Spanning Tree Protocol - Received None - Sent STP
   Portfast: off
   Configured Root guard: off; Operational Root guard: off
   Bpdu-guard: off
   Link-type: point-to-point
   Received BPDUs: 0; Sent BPDUs: 0
```

Observe that the settings comply with the formula set out in the STP parameters section, as:

(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

or in this case: 28 ≥ 20 ≥ 6.

13. Save the configuration.

```
device# copy running-config startup-config
```

Enable R-PVST+ on a system configuration example

```
device# configure terminal
device(config)# protocol spanning-tree rpvst
device(config-rpvst)# bridge-priority 4096
device(config-rpvst)# forward-delay  20
device(config-rpvst)# hello-time 8
device(config-rpvst)# max-age 22
device(config-rpvst)# transmit-holdcount 5
device(config-rpvst)# vlan 100 priority 0
device(config-rpvst)# vlan 201 priority 12288
device(config-rpvst)# vlan 301 priority 20480
device(config-rpvst)# interface ethernet 0/3
device(conf-if-eth-0/3)# switchport
device(conf-if-eth-0/3)# switchport mode trunk
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 100
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 201
device(conf-if-eth-0/3)# switchport trunk allowed vlan add 301
device(conf-if-eth-0/3)# no spanning-tree shutdown
device(conf-if-eth-0/3)# exit
device(config)# interface ethernet 0/4
device(conf-if-eth-0/4)# switchport
device(conf-if-eth-0/4)# switchport mode trunk
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 100
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 201
device(conf-if-eth-0/4)# switchport trunk allowed vlan add 301
device(conf-if-eth-0/4)# no spanning-tree shutdown
device(conf-if-eth-0/4)# end
device# show spanning-tree
device# copy running-config startup-config
```

Clearing spanning tree counters

Follow these steps to clear spanning tree counters on all interfaces or on the specified interface.

1. Clear spanning tree counters on all interfaces.

   ```
   device# clear spanning-tree counter
   ```

2. Clear spanning tree counters on a specified Ethernet interface.

   ```
   device# clear spanning-tree counter interface ethernet 0/3
   ```

3. Clear spanning tree counters on a specified port channel interface.

   ```
   device# clear spanning-tree counter interface port-channel 12
   ```

   Port channel interface numbers range from 1 through 64.

Clearing spanning tree-detected protocols

Follow these steps to restart the protocol migration process.

These commands force a spanning tree renegotiation with neighboring devices on either all interfaces or on a specified interface.

1. Restart the spanning tree migration process on all interfaces.

   ```
   device# clear spanning-tree detected-protocols
   ```

2. Restart the spanning tree migration process on a specific Ethernet interface.

   ```
   device# clear spanning-tree detected-protocols interface ethernet 0/3
   ```

3. Restart the spanning tree migration process on a specific port channel interface.

```
device# clear spanning-tree detected-protocols port-channel 12
```

   Port channel interface numbers range from 1 through 64.

Shutting down PVST+ or R-PVST+

Follow these steps to o shut down PVST+, or R-PVST+ either globally, on a specific interface, or a specific VLAN.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Shut down PVST+ or R-PVST+.
   - Shut down PVST+ or R-PVST+ globally and return to privileged EXEC mode.

```
device(config)# protocol spanning-tree pvst
device(config-pvst)# shutdown
device(config-pvst)# end
```

   - Shut down PVST+ or R-PVST+ on a specific interface and return to privileged EXEC mode.

```
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# spanning-tree shutdown
device(conf-if-eth-0/2)# end
```

   - Shut down PVST+ or R-PVST+ on a specific VLAN. and return to privileged EXEC mode.

```
device(config)# vlan 10
device(config-vlan-10)# spanning-tree shutdown
device(config-vlan-10)# end
```

3. Verify the configuration.

```
device# show spanning-tree
device#
```

4. Save the running configuration to the startup configuration.

```
device# copy running-config startup-config
```

Shut down PVST+ or R-PVST+ configuration example

```
device# configure terminal
device(config)# vlan 10
device(config-vlan-10)# spanning-tree shutdown
device(config-vlan-10)# end
device# show spanning-tree
device# copy running-config startup-config
```

NOTE

Shutting down PVST+ on a VLAN is used in this example.

802.1s Multiple Spanning Tree Protocol

MSTP overview

IEEE 802.1s Multiple STP (MSTP) helps create multiple loop-free active topologies on a single physical topology.

MSTP uses RSTP to group VLANs into separate spanning-tree instance. Each instance has its own spanning-tree topology independent of other spanning tree instances, which allows multiple forwarding paths, permits load balancing, and facilitates the movement of data traffic. A failure in one instance does not affect other instances. By enabling the MSTP, you are able to more effectively utilize the physical resources present in the network and achieve better load balancing of VLAN traffic.

The MSTP evolved as a compromise between the two extremes of the RSTP and R-PVST+, it was standardized as IEEE 802.1s and later incorporated into the IEEE 802.1Q-2003 standard. The MSTP configures a meshed topology into a loop-free, tree-like topology. When the link on a bridge port goes up, an MSTP calculation occurs on that port. The result of the calculation is the transition of the port into either a forwarding or blocking state. The result depends on the position of the port in the network and the MSTP parameters. All the data frames are forwarded over the spanning tree topology calculated by the protocol.

NOTE

Multiple switches must be configured consistently with the same MSTP configuration to participate in multiple spanning tree instances. A group of interconnected switches that have the same MSTP configuration is called an MSTP region.
MSTP is backward compatible with the STP and the RSTP.

Common Spanning Tree (CST)

The single Spanning Tree instance used by the Extreme device, and other vendor devices to interoperate with MSTP bridges. This spanning tree instance stretches across the entire network domain (including PVST, PVST+ and MSTP regions). It is associated with VLAN 1 on the Extreme device.

Internal Spanning Tree (IST)

An MSTP bridge must handle at least these two instances: one IST and one or more MSTIs (Multiple Spanning Tree Instances). Within each MST region, the MSTP maintains multiple spanning-tree instances. Instance 0 is a special instance known as IST, which extends CST inside the MST region. IST always exists if the device runs MSTP. Besides IST, this implementation supports up to 31 MSTIs.

Common Internal Spanning Tree (CIST)

The single spanning tree calculated by STP (including PVST+) and RSTP (including R-PVST+) and the logical continuation of that connectivity through MSTP bridges and regions, calculated by MSTP to ensure that all LANs in the bridged LAN are simply and fully connected

Multiple Spanning Tree Instance (MSTI)

One of a number of spanning trees calculated by the MSTP within an MST Region, to provide a simply and fully connected active topology for frames classified as belonging to a VLAN that is mapped to the MSTI by the MST configuration table used by the MST bridges of that MST region.

The Extreme implementation supports up to 32 spanning tree instances in an MSTP enabled bridge that can support up to 32 different Layer 2 topologies. The spanning tree algorithm used by the MSTP is the RSTP, which provides quick convergence.

By default all configured VLANs including the default VLAN are assigned to and derive port states from CIST until explicitly assigned to MSTIs.

MST regions

MST regions are clusters of bridges that run multiple instances of the MSTP protocol. Multiple bridges detect that they are in the same region by exchanging their configuration (instance to VLAN mapping), name, and revision-level. Therefore, if you need to have two bridges in the same region, the two bridges must have identical configurations, names, and revision-levels. Also, one or more VLANs can be mapped to one MST instance (IST or MSTI) but a VLAN cannot be mapped to multiple MSTP instances

MSTP regions

MSTP introduces a hierarchical way of managing device domains using regions. Devices that share common MSTP configuration attributes belong to a region. The MSTP configuration determines the MSTP region where each device resides. The common MSTP configuration attributes are as follows:

- Alphanumeric configuration name (32 bytes)
- Configuration revision number (2 bytes)
- 4096-element table that maps each of the VLANs to an MSTP instance

Region boundaries are determined by the above attributes. An MSTI is an RSTP instance that operates inside an MSTP region and determines the active topology for the set of VLANs mapping to that instance. Every region has a CIST that forms a single spanning tree instance which includes all the devices in the region. The difference between the CIST instance and the MSTP instance is that the CIST instance operates across the MSTP region and forms a loop-free topology across regions, while the MSTP instance operates only within a region. The CIST instance can operate using the RSTP only if all the devices across the regions support the RSTP. However, if any of the devices operate using the STP, the CIST instance reverts to the STP.

Each region is viewed logically as a single STP or a single RSTP bridge to other regions.

NOTE
Extreme supports 32 MSTP instances and one MSTP region.

For more information about spanning trees, see the introductory sections in the Spanning Tree Protocol chapter.

MSTP guidelines and restrictions

Follow these restrictions and guidelines when configuring the MSTP:

- Create VLANs before mapping them to the MSTP instances.
- The Extreme implementation of the MSTP supports up to 32 MSTP instances and one MSTP region.
- The MSTP **force-version** option is not supported.
- You must create VLANs before mapping them to the MSTP instances.

- For two or more switches to be in the same the MSTP region, they must have the same VLAN-to-instance map, the same configuration revision number, and the same region name.
- MSTP is backward compatible with the STP and the RSTP.
- Only one MSTP region can be configured on a bridge.
- A maximum of 4090 VLANs can be configured across the 32 MSTP instances.
- MSTP and topology groups cannot be configured together.
- MSTP configured over MCT VLANs is not supported.

Default MSTP configuration

As well as the defaults listed in the section Understanding the default STP configuration on page 201 there are defaults that apply only to MSTP configurations.

Parameter	Default setting
Cisco interoperability	Disabled
Device priority (when mapping a VLAN to an MSTP instance)	32768
Maximum hops	20 hops
Revision number	0

Interoperability with PVST+ and R-PVST+

Since Extreme or other vendor devices enabled with PVST+ and R-PVST+ send IEEE STP BPDUs in addition to the PVST and R-PVST BPDUs, the VLAN 1 spanning tree joins the Common Spanning Tree (CST) of the network and thus interoperates with MSTP. The IEEE compliant devices treat the BPDUs addressed to the Extreme proprietary multicast MAC address as an unknown multicast address and flood them over the active topology for the particular VLAN.

MSTP global level parameters

To configure a switch for MSTP, first you set the region name and the revision on each switch that is being configured for MSTP. You must then create an MSTP Instance and assign an ID. VLANs are then assigned to MSTP instances. These instances must be configured on all switches that interoperate with the same VLAN assignments.

Each of the steps used to configure and operate MSTP are described in the following:

NOTE
The MSTP Region and Revision global parameters are enabled for interface level parameters as described below.

- Set the MSTP region name — Each switch that is running MSTP is configured with a name. It applies to the switch which can have many different VLANs that can belong to many different MSTP regions. The default MSTP name is "NULL".
- Set the MSTP revision number — Each switch that is running MSTP is configured with a revision number. It applies to the switch, which can have many different VLANs that can belong to many different MSTP regions.
- Enabling and disabling Cisco interoperability — While in MSTP mode, use the **cisco-interoperability** command to enable or disable the ability to interoperate with certain legacy Cisco switches. If Cisco interoperability is required on any switch in the network, then all switches in the network must be compatible, and therefore enabled by means of this command. By default the Cisco interoperability is disabled.

- The parameters you would normally set when you configure STP are applicable to MSTP. Before you configure MSTP parameters see the sections explaining bridge parameters, the error disable timeout parameter and the port-channel path cost parameter in the STP section of this guide.

MSTP interface level parameters

Edge port and automatic edge detection

Configuring the edge port feature makes a port transition directly from initialization to the forwarding state, skipping the listening and learning states.

From an interface, you can configure a device to automatically identify the edge port. The port can become an edge port if no BPDU is received. By default, automatic edge detection is disabled.

Follow these guidelines to configure a port as an edge port:

- When edge port is enabled, the port still participates in a spanning tree.
- A port can become an edge port if no BPDU is received.
- When an edge port receives a BPDU, it becomes a normal spanning tree port and is no longer an edge port.
- Because ports that are directly connected to end stations cannot create bridging loops in the network, edge ports transition directly to the forwarding state and skip the listening and learning states.

  NOTE
  If BPDUs are received on a port fast enabled interface, the interface loses the edge port status unless it receives a **shutdown** or **no shutdown** command.

BPDU guard

In an STP environment, switches, end stations, and other Layer 2 devices use BPDUs to exchange information that STP will use to determine the best path for data flow.

In a valid configuration, edge port-configured interfaces do not receive BPDUs. If an edge port-configured interface receives a BPDU, an invalid configuration exists, such as the connection of an unauthorized device. The BPDU Guard provides a secure response to invalid configurations because the administrator must manually put the interface back in service.

BPDU guard removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

In some instances, it is unnecessary for a connected device, such as an end station, to initiate or participate in an STP topology change. In this case, you can enable the STP BPDU guard feature on the Extreme device port to which the end station is connected. The STP BPDU guard shuts down the port and puts it into an "error disabled" state. This disables the connected device's ability to initiate or participate in an STP topology. A log message is then generated for a BPDU guard violation, and a message is displayed to warn the network administrator of an invalid configuration.

The BPDU Guard provides a secure response to invalid configurations because the administrator must manually put the interface back in service with the **no shutdown** command if error disable recovery is not enabled by enabling the **errdisable-timeout** command. The interface can also be automatically configured to be enabled after a timeout. However, if the offending BPDUs are still being received, the port is disabled again.

Expected behavior in an interface context

When BPDU Guard is enabled on an interface, the device is expected to put the interface in Error Disabled state when BPDU is received on the port when edge-port and BPDU guard is enabled on the switch interface. When the port ceases to receive the BPDUs, it does not automatically switch to edge port mode, you must configure **error disable timeout** or **no shutdown** on the port to move the port back into edge port mode.

Restricted role

Configuring restricted role on a port causes the port not to be selected as root port for the CIST or any MSTI, even if it has the best spanning tree priority vector.

Restricted role ports are selected as an alternate port after the root port has been selected. It is configured by a network administrator to prevent bridges external to a core region of the network influencing the spanning tree active topology, possibly because those bridges are not under the full control of the administrator. It will protect the root bridge from malicious attack or even unintentional misconfigurations where a bridge device which is not intended to be root bridge, becomes root bridge causing severe bottlenecks in data path. These types of mistakes or attacks can be avoided by configuring 'restricted-role' feature on ports of the root bridge . This feature is similar to the "root-guard" feature which is proprietary implementation of Cisco for STP and RSTP but had been adapted in the 802.1Q standard as "restricted-role". The "restricted-role" feature if configured on an incorrect port can cause lack of spanning tree connectivity.

Expected behavior in an interface context

When this feature is enabled on an interface the device is expected to prevent a port configured with restricted-role feature from assuming the role of a Root port. Such a port is expected to assume the role of an Alternate port instead, once Root port is selected.

Restricted TCN

TCN BPDUs are used to inform other switches of port changes.

Configuring "restricted TCN" on a port causes the port not to propagate received topology change notifications and topology changes originated from a bridge external to the core network to other ports. It is configured by a network administrator to prevent bridges external to a core region of the network from causing MAC address flushing in that region, possibly because those bridges are not under the full control of the administrator for the attached LANs. If configured on an incorrect port it can cause temporary loss of connectivity after changes in a spanning trees active topology as a result of persistent incorrectly learned station location information.

Expected behavior in an interface context

When this feature is enabled on an interface, the device is expected to prevent propagation of topology change notifications from a port configured with the Restricted TCN feature to other ports. In this manner, the device prevents TCN propagation from causing MAC flushes in the entire core network.

Configuring MSTP

Enabling and configuring MSTP globally

Follow this procedure to configure the Multiple Spanning Tree Protocol.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable MSTP.

```
device(config)# protocol spanning-tree mstp
```

This command creates a context for MSTP. MSTP is automatically enabled. All MSTP specific CLI commands can be issued only from this context. Entering **no protocol spanning-tree mstp** deletes the context and all the configurations defined within the context.

3. Specify the region name.

```
device(config-mstp)# region kerry
```

4. Specify the revision number.

```
device(config-mstp)# revision 1
```

5. Configure an optional description of the MSTP instance.

```
device(config-mstp)# description kerry switches
```

6. Specify the maximum hops for a BPDU to prevent the messages from looping indefinitely on the interface.

```
device(config-mstp)# max-hops 25
```

Setting this parameter prevents messages from looping indefinitely on the interface. The range is 1 through 40 hops while the default is 20 .

7. Map VLANs to MSTP instances and set the instance priority.
   a) Map VLANs 7 and 8 to instance 1.

```
device(config-mstp)# instance 1 vlan 7,8
```

   b) Map VLANs 21, 22, and 23 to instance 2.

```
device(config-mstp)# instance 2 vlan 21-23
```

   c) Set the priority of instance 1.

```
device(config-mstp)# instance 1 priority 4096
```

This command can be used only after the VLAN is created. VLAN instance mapping is removed from the configuration if the underlying VLANs are deleted.

8. Configure a bridge priority for the CIST bridge.

```
device(config-mstp)# bridge-priority 4096
```

The range is 0 through 61440 in increments of 4096. The default is 32768.

9. Set the error disable parameters.
   a) Enable the timer to bring the port out of error disable state.

```
device(config-mstp)# error-disable-timeout enable
```

   b) Specify the time in seconds it takes for an interface to time out..

```
device(config-mstp)# error-disable-timeout interval 60
```

   The range is fro 10 to 1000000 seconds with a default of 300 seconds.
10. Configure forward delay.
   a) Specify the bridge forward delay.

```
device(config-mstp)# forward-delay 15
```

   This command allows you to specify how long an interface remains in the listening and learning states before it begins forwarding. This command affects all MSTP instances. The range of values is from 4 to 30 seconds with a default of 15 seconds.
11. Configure hello time.

```
device(config-mstp)# hello-time 2
```

   The hello time determines how often the switch interface broadcasts hello BPDUs to other devices. The range is from 1 through 10 seconds with a default of 2 seconds.
12. Configure the maximum age.

```
device(config-mstp)# max-age 20
```

   You must set the **max-age** so that it is greater than the **hello-time**. The range is 6 through 40 seconds with a default of 20 seconds.
13. Specify the port-channel path cost.

```
device(config-mstp)# port-channel path-cost custom
```

   This command allows you to control the path cost of a port channel according to bandwidth.
14. Specify the transmit hold count.

```
device(config-mstp)# transmit-holdcount 5
```

   The transmit hold count is used to limit the maximum number of MSTP BPDUs that the bridge can transmit on a port before pausing for 1 second. The range is from 1 to 10 seconds with a default of 6 seconds.
15. Configure Cisco interoperability.

```
device(config-mstp)# cisco-interoperability enable
```

   This command enables the ability to interoperate with certain legacy Cisco switches. The default is Cisco interoperability is disabled.
16. Return to privileged exec mode.

```
device(config-mstp)# end
```

17. Verify the configuration. The following is an example configuration.

```
device# show spanning-tree mst-config

Spanning-tree Mode: Multiple Spanning Tree Protocol

 CIST Root Id: 8000.001b.ed9f.1700
 CIST Bridge Id: 8000.768e.f80a.6800
 CIST Reg Root Id: 8000.001b.ed9f.1700

 CIST Root Path Cost: 0; CIST Root Port: Eth 1/2
 CIST Root Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 19
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20;
 Tx-HoldCount: 6
 Number of topology change(s): 139; Last change occurred 00:03:36 ago on Eth 1/2

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec
 Migrate Time: 3 sec

  Name           : kerry
  Revision Level : 1
  Digest         : 0x9357EBB7A8D74DD5FEF4F2BAB50531AA

  Instance        VLAN
  --------        ----
   0:             1
   1:             7,8
   2:             21-23
```

Observe that the settings comply with the formula set out in the STP parameters section, as:

(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

or in this case: 28 ≥ 20 ≥ 6.

18. Save the configuration.

```
device# copy running-config startup-config
```

MSTP configuration example

```
device# configure terminal
device(config)# protocol spanning-tree mstp
device(config-mstp)# region kerry
device(config-mstp)# revision 1
device(config-mstp)# description kerry switches
device(config-mstp)# max-hops 20
device(config-mstp)# instance 1 vlan 7,8
device(config-mstp)# instance 2 vlan 21-23
device(config-mstp)# instance 1 priority 4096
device(config-mstp)# bridge-priority 4096
device(config-mstp)# error-disable-timeout enable
device(config-mstp)# error-disable-timeout interval 60
device(config-mstp)# forward-delay 16
device(config-mstp)# hello-time 5
device(config-mstp)# max-age 16
device(config-mstp)# port-channel path-cost custon
device(config-mstp)# transmit-holdcount 5
device(config-mstp)# cisco-interoperability enable
device(config-mstp)# end
device# show spanning-tree mst
device# copy running-config startup-config
```

Enabling and configuring MSTP on an interface

Follow these steps to configure and enable MSTP on an Ethernet interface.

The parameters can be configured individually on an interface by:

1. Entering the commands in Steps 1 through Step 3 for the target interface
2. Running the relevant parameter command
3. Verifying the result
4. Saving the configuration

For detailed descriptions of the parameters and features, see the sections STP parameters and STP features.

1. Enter configuration mode.

```
device# configure terminal
```

2. Enable MSTP.

```
device(config)# protocol spanning-tree mstp
```

3. Enter interface configuration mode.

```
device(config-mstp)# interface ethernet 0/5
```

4. Enable the interface.

```
device(conf-if-eth-0/5)# no shutdown
```

5. Configure the restricted role feature for the port.

```
device(conf-if-eth-0/5)# spanning-tree restricted-role
```

   This command keeps a port from becoming a root.

6. Restrict topology change notifications (TCN) BPDUs for an MSTP instance.

```
device(conf-if-eth-0/5)# spanning-tree instance 5 restricted-tcn
```

   This prevents the port from propagating received TCNs and topology changes originating from a bridge, external to the core network, to other ports.

7. Enable auto detection of an MSTP edge port.

```
device(conf-if-eth-0/5)# spanning-tree autoedge
```

   Enabling this feature allows the system to automatically identify the edge port. The port can become an edge port if no BPDU is received. By default, automatic edge detection is disabled.

8.

```
device(conf-if-eth-0/5)# spanning-tree edgeport
```

   Enabling edge port allows the port to quickly transition to the forwarding state. By default, automatic edge detection is disabled.

9. Enable BPDU guard on the port

```
device(conf-if-eth-0/5)# spanning-tree edgeport bpdu-guard
```

   BPDU guard removes a node that reflects BPDUs back in the network. It enforces the STP domain borders and keeps the active topology predictable by not allowing any network devices behind a BPDU guard-enabled port to participate in STP.

10. Set the path cost of a port.

```
device(conf-if-eth-0/5)# spanning-tree cost 200000
```

    The path cost range is from 1 to 200000000. Leaving the default adjusts path cost relative to changes in the bandwidth. A lower path cost indicates greater likelihood of becoming root port.

11. Configure the link type.

```
device(conf-if-eth-0/5)# spanning-tree link-type point-to-point
```

    The options are point-to-point or shared.

12. Enable port priority.

```
device(conf-if-eth-0/5)# spanning-tree priority 128
```

    The range is from 0 to 240 in increments of 16 with a default of 32. A lower priority indicates greater likelihood of becoming root port.

13. Return to privileged exec mode.

```
device(conf-if-eth-0/5)# end
```

14. Verify the configuration.

```
device# show spanning-tree interface ethernet 0/5

Spanning-tree Mode: Multiple Spanning Tree Protocol

 Root Id: 8000.001b.ed9f.1700
 Bridge Id: 8000.01e0.5200.011d

 Port Eth 0/5 enabled
    Ifindex: 411271175; Id: 8002; Role: Designated; State: Forwarding
    Designated External Path Cost: 0; Internal Path Cost: 20000000
    Configured Path Cost: 200000
    Designated Port Id: 8002; Port Priority: 128
    Designated Bridge: 8000.01e0.5200.011d
    Number of forward-transitions: 1
    Version: Multiple Spanning Tree Protocol - Received MSTP - Sent MSTP
    Edgeport: yes; AutoEdge: yes; AdminEdge: no; EdgeDelay: 3 sec
    Restricted-role is enabled
    Restricted-tcn is enabled
    Boundary: no
    Bpdu-guard: on
    Link-type: point-to-point
    Received BPDUs: 86; Sent BPDUs: 1654
```

15. Save the configuration.

```
device# copy running-config startup-config
```

Enable MSTP on an interface configuration example

```
device# configure terminal
device(config)# protocol spanning-tree mstp
device(config-mstp)# interface ethernet 0/5
device(conf-if-eth-0/5)# no shutdown
device(conf-if-eth-0/5)# spanning-tree restricted-role
device(conf-if-eth-0/5)# spanning-tree instance 5 restricted-tcn
device(conf-if-eth-0/5)# spanning-tree autoedge
device(conf-if-eth-0/5)# spanning-tree edgeport
device(conf-if-eth-0/5)# spanning-tree edgeport bpdu-guard
device(conf-if-eth-0/5)# spanning-tree cost 200000
device(conf-if-eth-0/5)# spanning-tree link-type point-to-point
device(conf-if-eth-0/5)# spanning-tree priority 128
device(conf-if-eth-0/5)# end
device# show spanning-tree interface ethernet 0/5
device# copy running-config startup-config
```

Enabling MSTP on a VLAN

1. Enter configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the protocol command to enable MSTP configuration.

   ```
   device(config)# protocol spanning-tree mstp
   ```

3. Map a VLAN to an MSTP instance.

   ```
   device(config-mstp)# instance 5 vlan 300
   ```

4. Return to privileged EXEC mode.

   ```
   device(config-mstp)# end
   ```

5. Verify the configuration.

```
device# show spanning-tree mst

 Spanning-tree Mode: Multiple Spanning Tree Protocol

 CIST Root Id: 8000.609c.9f5d.4800 (self)
 CIST Bridge Id: 8000.609c.9f5d.4800
 CIST Reg Root Id: 8000.609c.9f5d.4800 (self)

 CIST Root Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20;
 Tx-HoldCount: 6
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec
 Migrate Time: 3 sec

  Name           : NULL
  Revision Level : 0
  Digest         : 0xD5FF4C3F6C18E2F27AF3A8300297ABAA

  Instance        VLAN
  --------        ----
   0:             1
   5:             100
```

   Observe that the settings comply with the formula set out in the STP parameters section, as:

   (2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))

   or in this case: 28 ≥ 20 ≥ 6.

6. Save the configuration.

```
device# copy running-config startup-config
```

Enable spanning tree on a VLAN configuration example

```
device# configure terminal
device(config)# protocol spanning-tree mstp
device(config-mstp)# instance 5 vlan 300
device(config-mstp)# end
device# show spanning-tree mst
device# copy running-config startup-config
```

Configuring basic MSTP parameters

Follow these steps to configure basic MSTP parameters.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable MSTP.

```
device(config)# protocol spanning-tree mstp
```

3. Specify the region name.

```
device(config-mstp)# region connemara
```

4. Specify the revision number.

```
device(config-mstp)# revision 1
```

5. Map MSTP instances to VLANs.
   a) Map instance 1 to VLANs 2 and 3.

```
device(config-mstp)# instance 1 vlan 2,3
```

   b) Map instance 2 to VLANs 4, 5, and 6.

```
device(config-mstp)# instance 2 vlan 4-6
```

6. Set a priority for an instance.

```
device(conf-Mstp)# instance 1 priority 28672
```

   The priority ranges from 0 through 61440 and the value must be in multiples of 4096.

7. Specify the maximum hops for a BPDU.

```
device(conf-Mstp)# max-hops 25
```

   This prevents the messages from looping indefinitely on an interface

8. Return to privileged EXEC mode.

```
device(conf-Mstp)# end
```

9. Verify the configuration.

```
device# show spanning-tree mst

 Spanning-tree Mode: Multiple Spanning Tree Protocol

 CIST Root Id: 8000.609c.9f5d.4800 (self)
 CIST Bridge Id: 8000.609c.9f5d.4800
 CIST Reg Root Id: 8000.609c.9f5d.4800 (self)

 CIST Root Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 20
 Configured Forward Delay: 15; Hello Time: 2; Max Age: 20; Max-hops: 25;
 Tx-HoldCount: 6
 Number of topology change(s): 0

 Bpdu-guard errdisable timeout: disabled
 Bpdu-guard errdisable timeout interval: 300 sec
 Migrate Time: 3 sec

  Name           : connemara
  Revision Level : 1
  Digest         : 0xD5FF4C3F6C18E2F27AF3A8300297ABAA

  Instance        VLAN
  --------        ----
   0:             1,7,8,9
   1:             2,3
   2:             4-6
```

NOTE

Observe that the settings comply with the formula set out in the STP parameters section, as:
(2 × (forward delay - 1)) ≥ maximum age ≥ (2 × (hello time + 1))
or in this case: 28 ≥ 20 ≥ 6.

```
device# show running-config | begin spanning-tree
protocol spanning-tree mstp
 instance 1 vlan 2,3
 instance 1 priority 28672
 instance 2 vlan 4-6
 region connemars
 revision 1
 max-hops 25
 !
...
```

10. Save the configuration

```
device# copy running-config startup-config
```

Basic MSTP configuration example

```
device# configure terminal
device(config)# protocol spanning-tree mstp
device(config-mstp)# region connemara
device(config-mstp)# revision 1
device(config-mstp)# instance 1 vlan 2,3
device(config-mstp)# instance 2 vlan 4-6
device(conf-Mstp)# instance 1 priority 28582
device(conf-Mstp)# max-hops 25
device(conf-Mstp)# end
device# show spanning-tree mst
device# copy running-config startup-config
```

Clearing spanning tree counters

Follow these steps to clear spanning tree counters on all interfaces or on the specified interface.

1. Clear spanning tree counters on all interfaces.

```
device# clear spanning-tree counter
```

2. Clear spanning tree counters on a specified Ethernet interface.

```
device# clear spanning-tree counter interface ethernet 0/3
```

3. Clear spanning tree counters on a specified port channel interface.

```
device# clear spanning-tree counter interface port-channel 12
```

   Port channel interface numbers range from 1 through 64.

Clearing spanning tree-detected protocols

Follow these steps to restart the protocol migration process.

These commands force a spanning tree renegotiation with neighboring devices on either all interfaces or on a specified interface.

1. Restart the spanning tree migration process on all interfaces.

```
device# clear spanning-tree detected-protocols
```

2. Restart the spanning tree migration process on a specific Ethernet interface.

```
device# clear spanning-tree detected-protocols interface ethernet 0/3
```

3. Restart the spanning tree migration process on a specific port channel interface.

```
device# clear spanning-tree detected-protocols port-channel 12
```

   Port channel interface numbers range from 1 through 64.

Shutting down MSTP

Follow these steps to o shut down MSTP either globally, on a specific interface, or a specific VLAN.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Shut down MSTP.
   - Shut down MSTP globally and return to privileged EXEC mode.

     ```
     device(config)# protocol spanning-tree mstp
     device(config-mstp)# shutdown
     device(config-mstp)# end
     ```

   - Shut down MSTP on a specific interface and return to privileged EXEC mode.

     ```
     device(config)# interface ethernet 0/2
     device(conf-if-eth-0/2)# spanning-tree shutdown
     device(conf-if-eth-0/2)# end
     ```

   - Shut down MSTP on a specific VLAN and return to privileged EXEC mode.

     ```
     device(config)# vlan 10
     device(config-vlan-10)# spanning-tree shutdown
     device(config-vlan-10)# end
     ```

3. Verify the configuration.

   ```
   device# show spanning-tree
   device#
   ```

4. Save the running configuration to the startup configuration.

   ```
   device# copy running-config startup-config
   ```

Shut down MSTP configuration example

```
device# configure terminal
device(config)# vlan 10
device(config-vlan-10)# spanning-tree shutdown
device(config-stp)# end
device# show spanning-tree
device# copy running-config startup-config
```

NOTE

Shutting down MSTP on a VLAN is used in this example.

Topology Groups

Topology Groups

A topology group is a named set of VLANs and bridge-domains that share a Layer 2 control protocol. Topology groups simplify configuration and enhance scalability of Layer 2 protocols by allowing you to run a single instance of a Layer 2 protocol on multiple VLANs and bridge-domains. One instance of the Layer 2 protocol controls all the VLANs and bridge-domains.

You can use topology groups with the following Layer 2 protocols:

- Per VLAN Spanning Tree (PVST+)
- Rapid per VLAN Spanning tree (R-PVST+)

Master VLAN, member VLANs, and bridge-domains

Each topology group contains a master VLAN and can contain one or more member VLANs and bridge-domains. A definition for each of these VLAN types follows:

- Master VLAN—The master VLAN contains the configuration information for the Layer 2 protocol. For example, if you plan to use the topology group for Rapid per VLAN Spanning tree (R-PVST), the topology group's master VLAN contains the R-PVST configuration information.
- Member VLANs—The member VLANs are additional VLANs that share ports with the master VLAN. The Layer 2 protocol settings for the ports in the master VLAN apply to the same ports in the member VLANs. A change to the master VLAN's Layer 2 protocol configuration or Layer 2 topology affects all the member VLANs. Member VLANs do not independently run a Layer 2 protocol.
- Member bridge domains—The member bridge domains are similar to VLANs that share ports with the master VLAN. The Layer 2 protocol settings for the ports in the master VLAN apply to the same ports in the bridge domains. A change to the master VLAN's Layer 2 protocol configuration or Layer 2 topology affects all the bridge domains. Bridge domains do not independently run a Layer 2 protocol. In a bridge domain, a single port can have multiple logical interfaces. In this scenario, all the logical interfaces on that port (and bridge domain) will follow the state of master VLAN port.

When a Layer 2 topology change occurs, resulting in a change of port state in the master VLAN, the same port state is applied to all the member VLANs and bridge-domains belonging to the topology group on that port. For example, if you configure a topology group whose master VLAN contains ports 1/1 and 1/2, a Layer 2 state change on port 1/1 applies to port 1/1 in all the member VLANs and bridge-domains that contain that port. However, the state change does not affect port 1/1 in VLANs that are not members of the topology group.

Control ports and free ports

A port in a topology group can be a control port or a free port:

- A **control port** is a port in the master VLAN and, therefore, is controlled by the Layer 2 protocol configured in the master VLAN. The same port in all the member VLANs and bridge-domains is controlled by the master VLAN's Layer 2 protocol. Each member VLAN and bridge-domain must contain all of the control ports. All other ports in the member VLAN and bridge-domain are "free ports."
- **Free ports** are not controlled by the master VLAN's Layer 2 protocol. The master VLAN can contain free ports. (In this case, the Layer 2 protocol is disabled on those ports.) In addition, any ports in the member VLANs and bridge-domains that are not also in the master VLAN are free ports.

  NOTE
  Because free ports are not controlled by the master port's Layer 2 protocol, they are always in the forwarding state.

Configuration considerations

The configuration considerations are as follows:

- You can configure up to 128 topology groups. A VLAN or bridge-domain cannot controlled by more than one topology group. You can configure up to 4K VLANs or bridge-domain as members of topology group.
- The topology group must contain a master VLAN. The group can also contain individual member VLANs and or member bridge-domains. You must configure the member VLANs or member bridge-domains before adding them to the topology group. Bridge-domains cannot be configured as a master VLAN.
- You cannot delete a master VLAN from the topology group when the member VLANs or bridge-domains are in the topology group.
- The control port membership must match the master VLAN when adding a member VLAN or member bridge-domain.
- If a VLAN enabled with the PVST+ or R-PVST+ protocol is added as a member VLAN of a topology group, the protocol is disabled. The member VLAN is added to the topology group. If the VLAN is removed from the topology group, the protocol is disabled, and you must enable the protocol if required.
- Enabling STP on an interface is only allowed if both master VLAN and member VLAN or bridge-domains are configured on the interface across all topology groups.
- You cannot remove the master VLAN or member VLAN or bridge-domains from an STP enabled interface.
- Topology group configuration is allowed only with PVST+ and R-PVST+ spanning tree configurations.

Configuring a topology group

Follow this procedure to configure a topology group. Extreme SLX devices support creating 128 topology groups in a system.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **topology-group** command to create a topology group at the global configuration level.

```
device(config)# topology-group 1
device(conf-topo-group-1)#
```

NOTE

The **no topology-group** command deletes an existing topology group.

Configuring a master VLAN

Follow this procedure to configure a master VLAN in a topology group.

Before configuring a master VLAN, you should have configured a topology group.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **topology-group** command to create a topology group at the global configuration level.

```
device(config)# topology-group 1
device(conf-topo-group-1)#
```

3. Enter the **master-vlan** command to configure a master VLAN in the topology group.

```
device(conf-topo-group-1)# master-vlan 100
```

NOTE

The **no master-vlan** command removes an existing master VLAN from the topology group.

Adding member VLANs

Follow this procedure to add member VLANs to a topology group. Member VLANs follow the master VLAN protocol states and also no L2 protocol will be running on the member VLANs.

Before adding a member VLAN, you should have created a topology group and configured the master VLAN for that group. The VLAN should not be part of any other topology group. All control ports of master VLAN must also be configured for the member VLAN.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **topology-group** command to create a topology group at the global configuration level.

```
device(config)# topology-group 1
device(conf-topo-group-1)#
```

3. Enter the **master-vlan** command to configure a master VLAN in the topology group.

```
device(conf-topo-group-1)# master-vlan 100
```

4. Enter the **member-vlan** command to add member VLANs to the topology group.

```
device(conf-topo-group-1)# member-vlan add 200-201
```

NOTE
The **member-vlan remove** command removes an existing member VLAN from the topology group.

```
device(conf-topo-group-1)# member-vlan remove 200
```

Adding member bridge-domains

Follow this procedure to add member bridge domains to a topology group. Member bridge domains follow the master VLAN protocol states and also no L2 protocol will be running on the bridge domains.

Before adding a bridge domain, you should have created a topology group and configured the master VLAN for that group. The bridge-domain should not be part of any other topology group. All control ports of master VLAN must also be configured for the member bridge-domain.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter the **topology-group** command to create a topology group at the global configuration level.

```
device(config)# topology-group 1
device(conf-topo-group-1)#
```

3. Enter the **master-vlan** command to configure a master VLAN in the topology group.

```
device(conf-topo-group-1)# master-vlan 100
```

4. Enter the **member-bridge-domain** command to add member bridge-domains to the topology group.

```
device(conf-topo-group-1)# member-bridge-domain add 300
```

NOTE
The **member-bridge-domain remove** command removes an existing member bridge-domain from the topology group.

```
device(conf-topo-group-1)# member-bridge-domain remove 1
```

The example adds 300 as member bridge-domain to the topology group.

```
device# configure terminal
device(config)# topology-group 1
device(conf-topo-group-1)# master-vlan 100
device(conf-topo-group-1)# member-bridge-domain add 300
```

Replacing a master VLAN

For replacing the existing master VLAN of a topology group, use the **master-vlan** command with the new master VLAN.

To avoid temporary loops when the master VLAN is replaced by another VLAN, the following recommendation is made:

- Control ports for both the old and the new master VLAN must match.
- The new master VLAN and the old master VLAN must have same ports in the blocking state to avoid the possibility of temporary loops.

If the recommendation is not followed, and a new master VLAN is configured with a different convergence, the configuration is still accepted.

NOTE
The master VLAN replacement is accepted if both the old and the new master VLANs are spanning-tree disabled.

Displaying topology group information

Follow the procedure to display topology group information for a specified group.

Before displaying the topology group information, you should have configured a topology group and defined the master VLAN.

Enter the **show topology-group** command to display the group information.

```
device# show topology-group 1
Topology Group 1
==============================
Master VLAN : 100
L2 Protocol: R-PVST
Member VLANs : 200 300
Member Bridge-domains: 10
Control Ports : eth 2/1, eth 2/2, po10
Free Ports : VLAN: 200 –eth 2/3, po11
Bridge-domain: 10 –eth 2/3.20, po11.10
```

The example displays information about topology group 1.

The **show running-config** command displays topology group configurations.

```
device# show running-config
topology-group 1
   master-vlan 100
   member-vlan add 200 300
   member-bridge-domain add 10
```

Loop Detection

LD protocol overview

The loop detection (LD) protocol is an Extreme proprietary protocol used to detect and break Layer 2 loops caused by misconfigurations, thereby preventing packet storms.

Layer 2 networks rely on learning and flooding to build their forwarding databases. Because of the flooding nature of these networks, any loops can be disastrous as they cause broadcast storms.

> ATTENTION
> The LD feature should be used only as a tool to detect loops in the network. It should not be used to replace other Layer 2 protocols such as STP.

This feature provides support for the following:

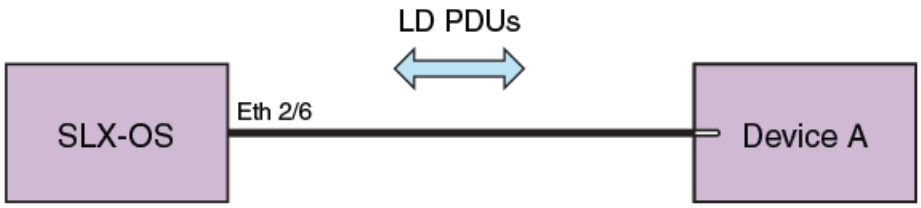
- Strict and loose modes
- Multi-Chassis Trunk (MCT)
- Breakout ports

LD protocol data units (PDUs) are initiated and received on the native device. Loop detection and action on the port state is also done on the same native device. Intermediate devices in the network must be capable of flooding unknown Layer 2 unicast PDUs on the VLAN through which they are received.

Strict mode

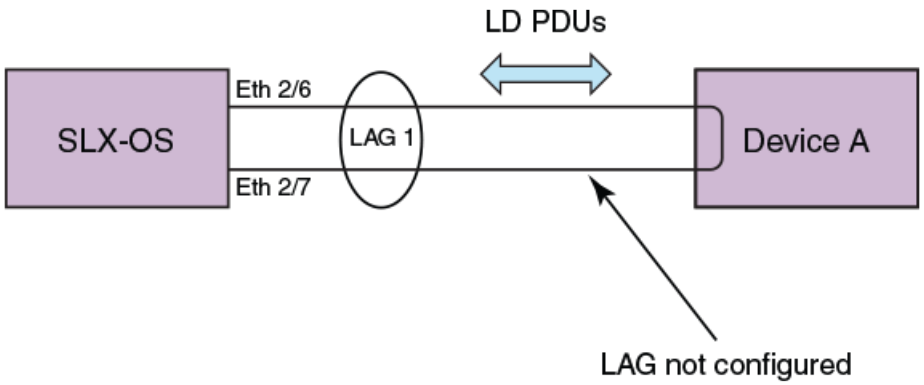
In what is referred to as *strict mode*, LD is configured on an interface. If the LD PDU is sent on an interface and received on the same interface, that port is shut down by LD. Strict mode overcomes specific hardware issues that cause packets to be echoed back to the input port. The following figure illustrates strict mode.

FIGURE 38 Strict mode



If the user provides a VLAN, then the PDUs are tagged accordingly. Otherwise PDUs are sent untagged. With a LAG, PDUs are sent out on the port-channel interface. If Device A has a loop (for example, a LAG is not configured), then the PDU is flooded back to SLX-OS, which detects the loop. In case of a loop, the port-channel interface is shut down. The following figure illustrates LD on a LAG.

FIGURE 39 LD on a LAG

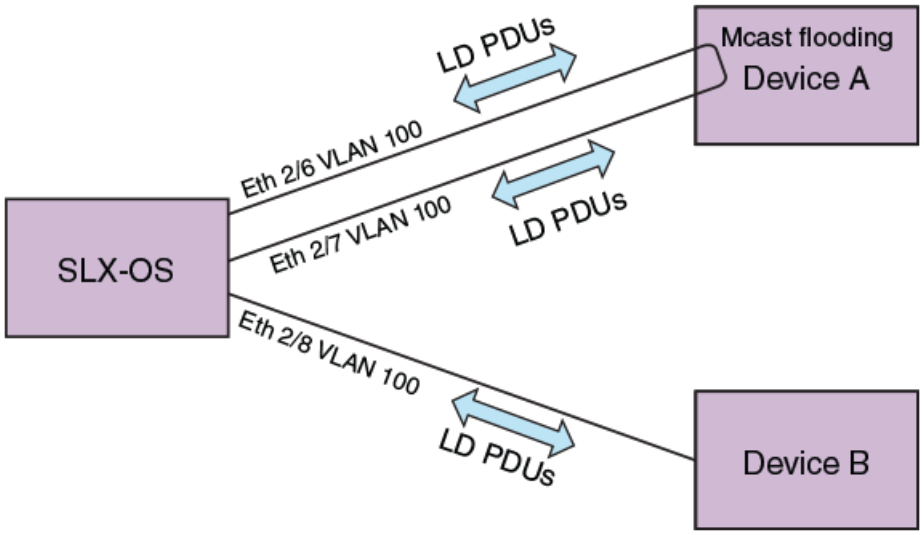


LD supports 256 instances of strict mode.

Loose mode

In what is referred to as *loose mode*, LD is configured on a VLAN. If a VLAN in the device receives an LD PDU that originated from the same device on that VLAN, this is considered to be a loop and the receiving port is shut down. In loose mode, LD works at the VLAN level and takes action at the link level. The following figure illustrates loose mode, with LD on a VLAN.

FIGURE 40 Loose mode: LD on a VLAN



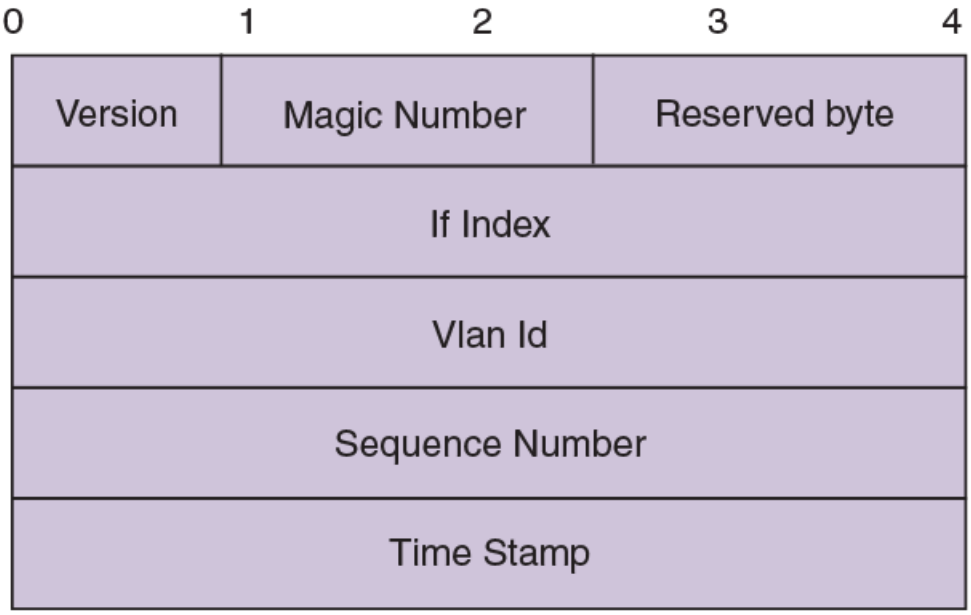
SLX-OS generates the LD PDUs on the VLAN. if Device A has a loop, PDUs are flooded back to SLX-OS, which detects the loop. SLX-OS then shuts down the receiving port on the VLAN.

LD supports 256 instances of loose mode, which means that it can be enabled on 256 VLANs.

LD PDU format

The following figure illustrates the format of the LD PDU in bytes.

FIGURE 41 LD PDU format in bytes



Parameter	Definition
Version	LD protocol version (1 by default)
Magic Number	0x13EF; used to differentiate between LD multicast PDUs and other multicast PDUs
Reserved byte	For future use
If Index	Index of the source port; populated only in strict mode
Vlan Id	VLAN ID
Sequence Number	Reserved for future enhancements
Time Stamp	Reserved for future enhancements

LD PDU transmission

Each LD-enabled interface or VLAN on a device continually transmits Layer 2 LD PDUs at a 1-second default hello-timer interval, with the destination MAC address as the multicast address. The multicast MAC address is derived from the system MAC address of the device with the multicast bit (8) and the local bit (7) set.

For example, if the MAC address is 00E0.5200.1800, then the multicast MAC address is 03E0.5200.1800. In the case of a LAG port-channel, LD PDUs are sent out one of the ports of the LAG as chosen by hardware.

LD PDU reception

When the LD PDU is received and is generated by the same device, the PDU is processed. If the PDU is generated by another device, then the PDU is flooded.

If a port is already blocked by any other Layer 2 protocol such as STP, then the LD PDUs are neither sent for LD processing nor flooded on that port.

LD parameters

This section discusses the various global protocol-level, interface level, and VLAN-level parameters that are used to control and process LD PDUs.

Protocol level

hello-interval

hello-interval is the rate at which the LD PDUs are transmitted by an LD-enabled interface or VLAN, which is 1000 milliseconds by default. Lowering the hello-interval below the default increases the PDU transmission rate, providing faster loop detection and also removing transient loops that last less than one second. On the other hand, increasing the interval above the default (for example, to 100 milliseconds) can increase the steady-state CPU load.

shutdown-time

shutdown-time is the duration after which an interface that is shut down by LD is automatically reenabled. The range is from 0 through 1440 minutes. The default is 0 minutes, which means that the interface is not automatically reenabled.

> ATTENTION
> Changing this value can cause repeated interface flapping when a loop is persistent in the network.

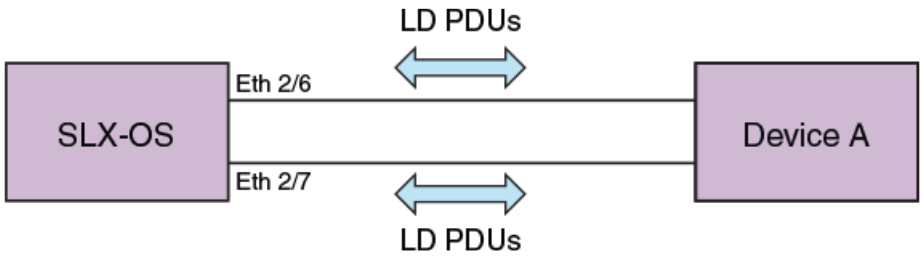
raslog-duration

raslog-duration is the interval between RASLog messages when a port is shut down by LD to prevent flooding of these messages. The range is from 10 through 1440 minutes. The default is 10.

Interface level

In strict mode, the parameters in this section are configurable at the interface level, and the configuration is specific to an interface. The following figure illustrates strict mode configuration.

FIGURE 42 Strict mode configuration



shutdown-disable

By default, the device shuts down the interface if a loop is detected. Configuring **shutdown-disable** means that the interface shutdown is disabled and LD never brings down such interface. If a loop is already detected by LD and the port is in shutdown state, then configuring **shutdown-disable** is not effective until the port is back up.

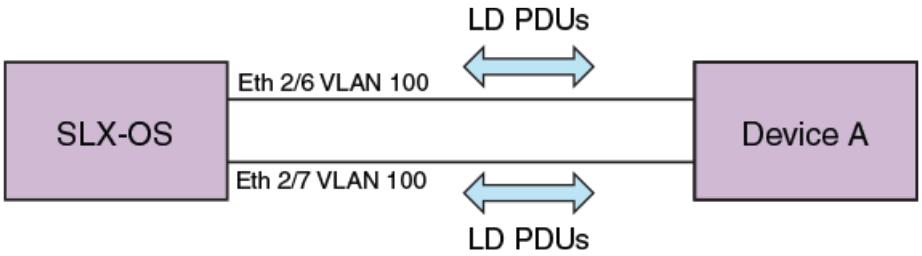
vlan-association

Although user can enable LD on an interface without specifying a VLAN, the **vlan-association** keyword is used to specify a VLAN associated with the interface.

VLAN level

In loose mode, the user can configure LD under a VLAN. In this case, LD PDUs are flooded on the VLAN. The following figure illustrates loose mode configuration.

FIGURE 43 Loose mode configuration



LD PDU processing

As long as LD PDUs are not received, there is no loop. If an LD PDU is received, then there is a loop that is present in the network.

If the if-index field in the received LD PDU is valid, then it is considered to be operating in strict mode. If the port on which the LD PDU was received is same as one encoded in the PDU (with a match for VLAN ID if a VLAN is associated), the port is shut down. For an MCT, if a strict mode LD PDU is received on an ICL interface, and the PDU is originated by another interface, then the ICL interface is not shut down. Instead, the sender interface is shut down. In addition, for strict mode the required interfaces should be configured with LD, or else the PDUs will not get processed

If the if-index field in the received LD PDU is invalid, then it is considered to be operating in loose mode. Based on VLAN ID information present in the received LD PDU, the receiving interface is shut down. If the receiving interface is an MCT ICL interface, the LD PDU is dropped.

In the case of a LAG (port-channel) interface, if the sent LD PDU is received on the port-channel, then the port-channel interface is shut down.

If the **shutdown-disable** option is configured for the particular interface, then the port drops the received PDU without processing it.

The re-enablement of the LD shut down port depends on the **shutdown-time** configuration. For manual recovery, either flap the port or clear the loop through the command line.

Configuration considerations

On an external switch that is unaware of LD or where LD is not configured, there may be some ACL rules applied to interfaces to permit traffic from known MAC addresses, and at the last of these rules there is an ACL deny-any rule to block all unknown MAC addresses. If this interface is part of a loop, LD enabled on SLX-OS will not be able to detect and break the loop.

LD use cases

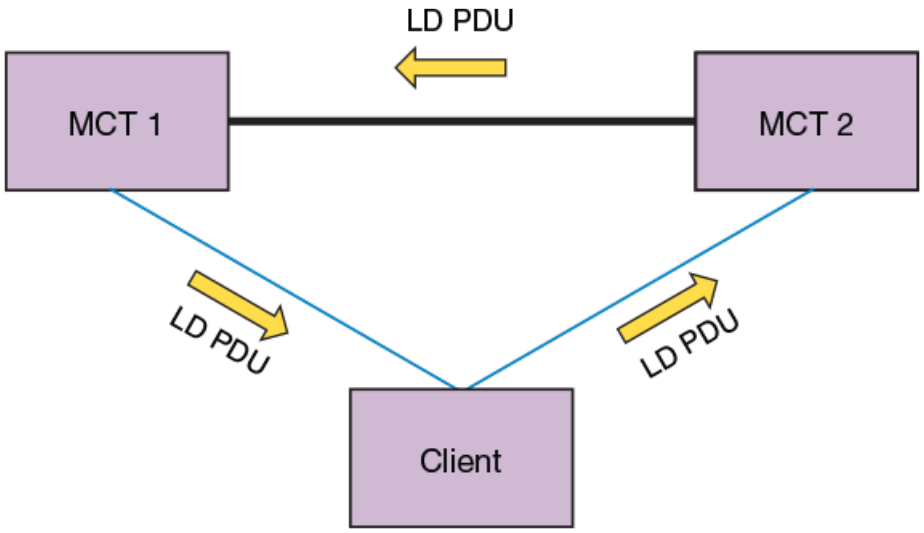
In an MCT configuration, LD runs independently on both nodes. With loose mode the user must enable loop detection for the same VLANs on both nodes in the MCT cluster. MCT strict mode and loose mode use cases are detailed below.

MCT strict mode

The following figure illustrates a use case for MCT strict mode, followed by a sequence of events.

Strict mode LD is enabled on the MCT 1 cluster client edge port (CCEP) interface that connects to the Client.

FIGURE 44 MCT strict mode

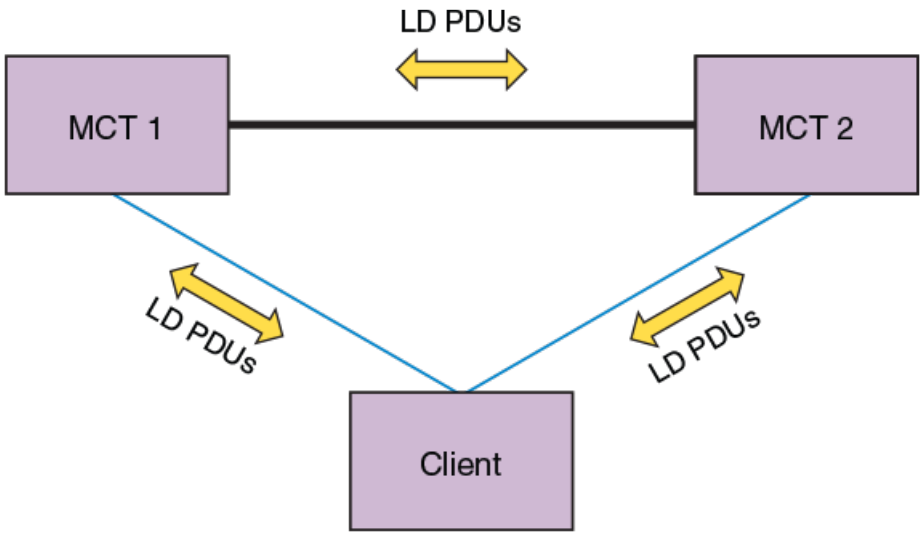


1. MCT 1 generates LD PDUs.
2. If the Client has the LAG interface configured to support LD, the Client drops the PDUs and there is no loop.
3. If there is a misconfiguration, the Client floods the PDUs, reaching MCT 2.
4. MCT 1 then identifies the interface information encoded in the PDUs, shutting down the interface on which the packets were generated.

MCT loose mode

The following figures illustrates two use cases for MCT loose mode, followed by a sequence of events.

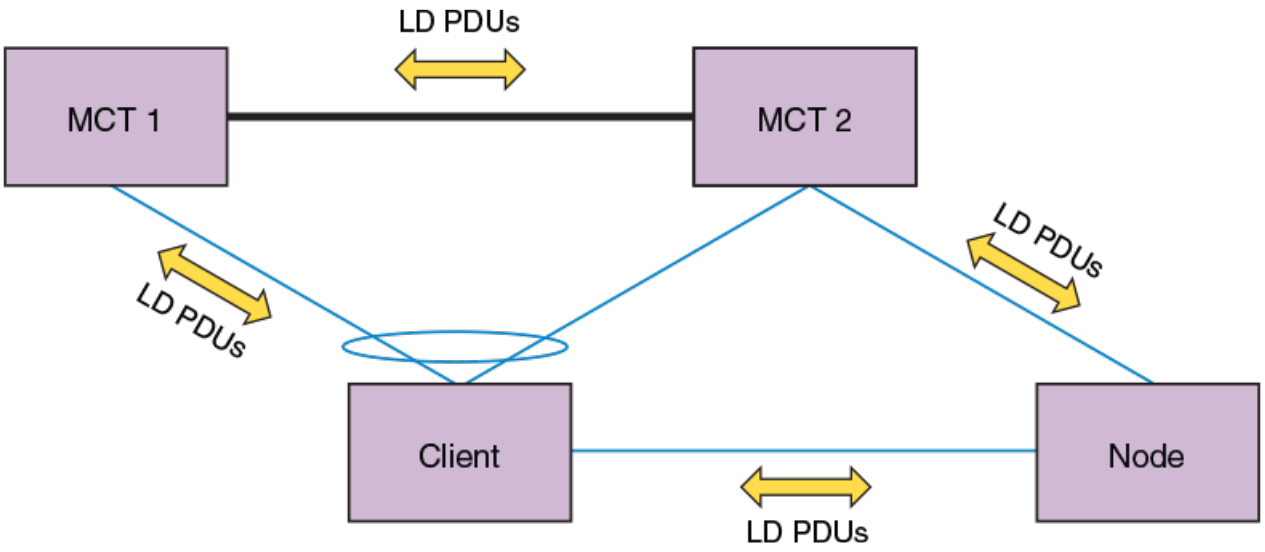
FIGURE 45 MCT loose mode: Use case 1



Use case 1: LD enabled on VLAN *x* on MCT 1

1. MCT 1 sends LD PDUs on VLAN x on all the interfaces that are part of the CCEP, client edge port (CEP), and ICL interface.
2. If the Client has LD configured on the LAG interface, then it drops the PDUs and no loop exists. If there is a misconfiguration, the Client floods the PDUs and they reach MCT 2.
3. MCT 2 floods the PDUs back to MCT 1, where the loop is detected. With loose mode no information about the interface that transmitted the PDU is encoded in the PDU, so normally the receiving interface is shut down. Because in this case the PDU is received on the ICL interface, that interface is not shut down.
4. MCT 1 receives the loop detection PDUs on the CCEP interface as well, as the packets were flooded in the VLAN in the following sequence: MCT 1 > MCT 2 > Client > MCT 1. In this case the receiving CCEP is shut down to break the loop. For MCT 2 to forward the PDUs in this case it must be the designated forwarder (DF) for that VLAN.

FIGURE 46 MCT loose mode: Use case 2



Use case 2: LD enabled on VLAN *x* on MCT 1 and MCT 2

1. Both MCT 1 and MCT 2 will flood the PDUs in VLAN *x* on all the interfaces that are part of the CCEP, CEP, and ICL interface.
2. Assuming PDUs from MCT 1 take the path MCT 1 > MCT 2 > Node > Client > MCT 1, then the receiving CCEP interface is shut down. For MCT 2 to forward the PDUs in this case, it must be the DF for that VLAN.
3. Assuming PDUs from MCT 2 take the path MCT 2 > MCT 1 > Client > Node > MCT 2, then the receiving CEP interface is shut down
4. If PDUs from MCT 2 take the path MCT 2 > Node > Client > MCT 2, then the receiving CCEP interface is shut down.
5. Multiple interfaces can be shut down in this case, depending on the sequence in which loops are detected.
6. In addition, to avoid CCEP interfaces from being shut down over a CEP interface, thje user can configure a CCEP port not to be shut down.

Configuring LD protocol

Follow these steps to configure loop detection (LD) protocol globally and at the interface and VLAN level.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **protocol loop-detection** command to enable loop detection, enter Protocol Loop Detect configuration mode, and configure a variety of global options.

```
device(config)# protocol loop-detection
```

3. (Optional) Enter the **hello-interval** command to change the hello interval from the default.

```
device(config-loop-detect)# hello-interval 2000
```

4. (Optional) Enter the **shutdown-time** command to change from the default the interval after which an interface that is shut down by loop detection (LD) protocol is automatically reenabled.

```
device(config-loop-detect)# shutdown-tiome 20
```

5. (Optional) Enter the **raslog-duration** command to change from default the interval between RASLog messages that are sent when a port is disabled by the loop detection (LD) protocol.

```
device(config-loop-detect)# raslog-duration 20
```

6. Enable LD at the interface level.
   a) In global configuration mode, specify an interface (either an Ethernet interface or a port-channel interface).

```
device(config)# interface ethernet 2/6
```

   b) In interface subtype configuration mode, enter the **loop-detection** command.

```
device(conf-if-eth-2/6)# loop-detection
```

7. Enable LD at the VLAN level.
   a) In global configuration mode, create a VLAN.

```
device(config)# vlan 5
```

   b) In VLAN configuration mode, enter the **loop-detection** command.

```
device(config-vlan-5)# loop-detection
```

8. Associate the VLAN with an interface.
   a) In global configuration mode, specify an interface (either an Ethernet interface or a port-channel interface).

```
device(config)# interface ethernet 2/6
```

   b) In interface subtype configuration mode, enter the **loop-detection vlan** command and specify a VLAN. (The VLAN must already be created.)

```
device(conf-if-eth-2/6)# loop-detection vlan 5
```

9. (Optional) Disable the shutting down of an interface (Ethernet or port-channel) as a result of the loop detection (LD) protocol.
   a) In global configuration mode, specify an interface (either an Ethernet interface or a port-channel interface).

```
device(config)# interface ethernet 2/6
```

   b) In interface subtype configuration mode, enter the **loop-detection shutdown-disable** command.

```
device(conf-if-eth-2/6)# loop-detection shutdown-disable
```

10. (Optional) Disable the shutting down of an interface (Ethernet or port-channel) as a result of the loop detection (LD) protocol.
    a) In global configuration mode, specify an interface (either an Ethernet interface or a port-channel interface).

```
device(config)# interface ethernet 2/6
```

    b) In interface subtype configuration mode, enter the **loop-detection shutdown-disable** command.

```
device(conf-if-eth-2/6)# loop-detection shutdown-disable
```

11. Confirm the LD configuration, using the **show loop-detection** command with a variety of options.
    a) To display LD information at the system level, enter the **show loop-detection** command as in the following example.

```
device# show loop-detection
Strict Mode:
------------------------

Number of loop-detection instances enabled: 1

Interface: eth 2/6
        Enabled on VLANs: 100
        Shutdown Disable:   No
        Interface status: UP
        Auto enable in: Never

Packet Statistics:
vlan            sent            rcvd        disable-count
100             100             0           0

Loose Mode:
------------------------

Number of LD instances:   2
Disabled Ports:                  2/7

Packet Statistics:
vlan            sent            rcvd        disable-count
100             100             0           0
```

    b) To display ports disabled by LD, enter the **show loop-detection disabled-ports** command as in the following example.

```
device# show loop-detection disabled-ports
Ports disabled by loop detection
-----------------------------------------
port       age(min)       disable cause
2/6        5              Disabled by Self
```

    c) To display global LD configuration values, enter the **show loop-detection globals** command.

```
device# show loop-detection globals
Loop Detection:                 Disabled
Shutdown-time (minutes):        0
Hello-time (msec):              1000
Raslog-duration (minutes):      10
```

12. Use the **clear loop-detection** command in privileged EXEC mode with a variety of options to reenable ports that were disabled by LD and clear the LD statistics.
    a) To enable LD-disabled ports and clear LD statistics on all interfaces, enter the **clear loop-detection** command.

```
device# clear loop-detection
```

    b) To enable LD-disabled ports and clear LD statistics on an Ethernet interface, enter the **clear loop-detection interface ethernet** command.

```
device# clear loop-detection interface ethernet 2/6
```

    c) To enable LD-disabled ports and clear LD statistics on a port-channel interface, enter the **clear loop-detection interface port-channel** command.

```
device# clear loop-detection interface port-channel 20
```

    d) To enable LD-disabled ports and clear LD statistics on a VLAN, enter the **clear loop-detection interface vlan** command.

```
device# clear loop-detection interface vlan 10
```

Loop detection for VLAN

LD loose mode is used to support a shutdown at the attachment circuit (AC) logical interface (LIF) level instead of at the physical port level.

When a loop is detected on a VLAN and port, only the LIF of the VLAN on the port is shut down, but the physical port still remains up and other VLANs on the port are not affected.

Configuring loop detection for VLAN

This section presents a variety of examples that configure Layer 2 loop detection for VLAN.

The following example enables loop detection on a VLAN and enters Protocol Loop Detection configuration mode.

```
device# configure terminal
device(config)# vlan 5
device(config-vlan-5)# loop-detection
device(config-loop-detect)#
```