Implementing Virtual Private LAN Services on Cisco IOS XR Software

This module provides the conceptual and configuration information for Virtual Private LAN Services (VPLS) on Cisco IOS XR software. VPLS supports Layer 2 VPN technology and provides transparent multipoint Layer 2 connectivity for customers.

This approach enables service providers to host a multitude of new services such as broadcast TV, Layer 2 VPNs, and so forth.

For MPLS Layer 2 virtual private networks (VPNs), see Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software module.

Note For more information about MPLS Layer 2 VPN on Cisco IOS XR software and for descriptions of the commands listed in this module, see the "Related Documents" section. To locate documentation for other commands that might appear while executing a configuration task, search online in the Cisco IOS XR software master command index.

Feature History for Implementing Virtual Private LAN Services on Cisco IOS XR Configuration Module

Release	Modification
Release 3.7.0	This feature was introduced on the Cisco XR 12000 Series Router.

Contents

Prerequisites for Implementing Virtual Private LAN Services on Cisco IOS XR Software

Before you configure VPLS, ensure that the network is configured as follows:

- Configure IP routing in the core so that the provider edge (PE) routers can reach each other through IP.
- Configure MPLS and Label Distribution Protocol (LDP) in the core so that a label switched path (LSP) exists between the PE routers.
- Configure a loopback interface to originate and terminate Layer 2 traffic. Make sure that the PE routers can access the other router's loopback interface.

Note The loopback interface is not needed in all cases. For example, tunnel selection does not need a loopback interface when VPLS is directly mapped to a TE tunnel.

You must be in a user group associated with a task group that includes the proper task IDs for MPLS L2VPN commands. For detailed information about user groups and task IDs, see the *Configuring AAA Services on Cisco IOS XR Software* module in the *Cisco IOS XR System Security Configuration Guide*.

Restrictions for Implementing Virtual Private LAN Services on Cisco IOS XR Software

The following Engine 3 hardware restrictions are listed for implementing VPLS:

- All attachment circuits in a bridge domain on an Engine 3 line card must be the same type (for example, port, dot1q, qinq, or qinany), value (VLAN ID), and EtherType (for example, 0x8100, 0x9100, or 0x9200).
- The same line card cannot simultaneously have attachment circuits and MPLS-enabled on any one of its interfaces. The line card cannot be Edge-facing and Core-facing at the same time.
- The line card requires ternary content addressable memory (TCAM) Carving configuration.

For the Engine 5 line card, version 1 of the Ethernet SPA does not support QinQ mode and QinAny mode.

Note For the Engine 5 line card, version 2 of the Ethernet SPA supports all VLAN modes, such as VLAN mode, QinQ mode, or QinAny mode.

Information About Implementing Virtual Private LAN Services

To implement Virtual Private LAN Services (VPLS), you should understand the following concepts:

- Virtual Private LAN Services Overview, page MPC-243
- VPLS for an MPLS-based Provider Core, page MPC-243
- Signaling, page MPC-244
- Bridge Domain, page MPC-244

Virtual Private LAN Services Overview

Virtual Private LAN Service (VPLS) enables geographically separated local-area network (LAN) segments to be interconnected as a single bridged domain over an MPLS network. The full functions of the traditional LAN such as MAC address learning, aging, and switching are emulated across all the remotely connected LAN segments that are part of a single bridged domain. A service provider can offer VPLS service to multiple customers over the MPLS network by defining different bridged domains for different customers. Packets from one bridged domain are never carried over or delivered to another bridged domain, thus ensuring the privacy of the LAN service.

VPLS transports Ethernet 802.3, VLAN 802.1q, and VLAN-in-VLAN (Q-in-Q) traffic across multiple sites that belong to the same Layer 2 broadcast domain. VPLS offers simple Virtual LAN services that include flooding broadcast, multicast, and unknown unicast frames that are received on a bridge. The VPLS solution requires a full mesh of pseudowires that are established among provider edge (PE) routers. The VPLS implementation is based on Label Distribution Protocol (LDP)-based pseudowire signaling.

VPLS is based on the characteristic of virtual forwarding instance (VFI). A VFI is a virtual bridge port that is capable of performing native bridging functions, such as forwarding, based on the destination MAC address, source MAC address learning and aging, and so forth.

After provisioning attachment circuits, neighbor relationships across the MPLS network for this specific instance are established through a set of manual commands identifying the end PEs. When the neighbor association is complete, a full mesh of pseudowires is established among the network-facing provider edge devices, which is a gateway between the MPLS core and the customer domain.

Now, the service provider network starts switching the packets within the bridged domain specific to the customer by looking at destination MAC addresses. All traffic with unknown, broadcast, and multicast destination MAC addresses is flooded to all the connected customer edge devices, which connect to the service provider network. The network-facing provider edge devices learn the source MAC addresses as the packets are flooded. The traffic is unicasted to the customer edge device for all the learned MAC addresses.

VPLS requires the provider edge device to be MPLS-capable. The VPLS provider edge device holds all the VPLS forwarding MAC tables and Bridge Domain information. In addition, it is responsible for all flooding broadcast frames and multicast replications.

VPLS for an MPLS-based Provider Core

VPLS is a multipoint Layer 2 VPN technology that connects two or more customer devices using bridging techniques. The VPLS architecture allows for the end-to-end connection between the Provider Edge (PE) routers to provide Multipoint Ethernet Services.

From an end-user perspective, VPLS emulates an Ethernet switch. VPLS requires the creation of a bridge domain (Layer 2 broadcast domain) on each of the PE routers. The access connections to the bridge domain on a PE router are called attachment circuits.

The attachment circuits can be a set of physical ports, virtual ports, or both that are connected to the bridge at each PE device in the network.

The MPLS/IP provider core simulates a virtual bridge that connects the multiple attachment circuits on each of the PE devices together to form a single broadcast domain. This also requires all of the PE routers that are participating in a VPLS instance to form emulated VCs among them.

A VFI is created on the PE router for each VPLS instance. The PE routers make packet-forwarding decisions by looking up the VFI of a particular VPLS instance. The VFI acts like a virtual bridge for a given VPLS instance. More than one attachment circuit belonging to a given VPLS are connected to the VFI. The PE router establishes emulated VCs to all the other PE routers in that VPLS instance and attaches these emulated VCs to the VFI. Packet forwarding decisions are based on the data structures maintained in the VFI.

Signaling

An important aspect of VPN technologies, including VPLS, is the ability of network devices to automatically signal to other devices about an association with a particular VPN, often referred to as *signaling mechanisms*. For VPLS, this includes discovery of other peers and MAC address withdrawal.

Note Border Gateway Protocol (BGP) auto-discovery and signaling are not supported.

The implementation of VPLS in a network requires the establishment of a full mesh of pseudowires between the provider edge (PE) routers. The signaling of pseudowires between provider edge devices, described in *draft-ietf-l2vpn-vpls-ldp-09*, uses targeted LDP sessions to exchange label values and attributes and to setup the pseudowires. LDP is an efficient mechanism for signaling pseudowire status for Ethernet point-to-point and multipoint services.

Bridge Domain

The native bridge domain refers to a Layer 2 broadcast domain consisting of a set of physical or virtual ports (including VFI). Data frames are switched within a bridge domain based on the destination MAC address. Multicast, broadcast, and unknown destination unicast frames are flooded within the bridge domain. In addition, the source MAC address learning is performed on all incoming frames on a bridge domain. A learned address is aged out. Incoming frames are mapped to a bridge domain, based on either the ingress port or a combination of both an ingress port and a MAC header field.

By default, split horizon is enabled on a bridge domain. In other words, any packets that are coming on either the attachment circuits or pseudowires are not returned on the same attachment circuits or pseudowires. In addition, the packets that are received on one pseudowire are not replicated on other pseudowires in the same VFI.

MAC Address-related Parameters

The MAC address table contains a list of the known MAC addresses and their forwarding information. In the current VPLS design, the MAC address table and its management are distributed. In other words, a copy of the MAC address table is maintained on the Route Processor (RP) card and the line cards. The RP card manages the master-copy of the MAC table, and is responsible to insert or delete the MAC addresses from the table and to distribute the new information to all line cards.

These topics provide information about the MAC address-related parameters:

- MAC Address Flooding, page MPC-245
- MAC Address-based Forwarding, page MPC-245
- MAC Address Source-based Learning, page MPC-245
- MAC Address Aging, page MPC-245
- MAC Address Limit, page MPC-246
- MAC Address Withdrawal, page MPC-246

MAC Address Flooding

Ethernet services require that frames that are sent to broadcast addresses and to unknown destination addresses be flooded to all ports. To obtain flooding within VPLS broadcast models, all unknown unicast, broadcast, and multicast frames are flooded over the corresponding pseudowires and to all attachment circuits. Therefore, a PE must replicate packets across both attachment circuits and pseudowires.

MAC Address-based Forwarding

To forward a frame, a PE must associate a destination MAC address with a pseudowire or attachment circuit. This type of association is provided through a static configuration on each PE or through dynamic learning, which is flooded to all bridge ports.

Note Split horizon forwarding applies in this case, for example, frames that are coming in on an attachment circuit or pseudowire are sent out of the same pseudowire. The pseudowire frames, which are received on one psuedowire, are not replicated on other pseudowires in the same virtual forwarding instance (VFI).

MAC Address Source-based Learning

When a frame arrives on a bridge port (for example, pseudowire or attachment circuit) and the source MAC address is unknown to the receiving PE router, the source MAC address is associated with the pseudowire or attachment circuit. Outbound frames to the MAC address are forwarded to the appropriate pseudowire or attachment circuit.

MAC address source-based learning uses the MAC address information that is learned in the hardware forwarding path. The updated MAC tables are sent to all line cards (LCs) and program the hardware for the router.

The number of learned MAC addresses is limited through configurable per-port and per-bridge domain MAC address limits.

MAC Address Aging

A MAC address in the MAC table is considered valid only for the duration of the MAC address aging time. When the time expires, the relevant MAC entries are repopulated. When the MAC aging time is configured only under a bridge domain, all the pseudowires and attachment circuits in the bridge domain use that configured MAC aging time.

A bridge forwards, floods, or drops packets based on the bridge table. The bridge table maintains both static entries and dynamic entries. Static entries are entered by the network manager or by the bridge itself. Dynamic entries are entered by the bridge learning process. A dynamic entry is automatically removed after a specified length of time, known as *aging time*, from the time the entry was created or last updated.

If hosts on a bridged network are likely to move, decrease the aging-time to enable the bridge to adapt to the change quickly. If hosts do not transmit continuously, increase the aging time to record the dynamic entries for a longer time, thus reducing the possibility of flooding when the hosts transmit again.

MAC Address Limit

The MAC address limit is used to limit the number of learned MAC addresses. The limit is set at the bridge domain level and the port level. When the MAC address limit is violated, the system is configured to take one of the actions that are listed in Table 5.

Table 5 MAC Address Limit Actions

Action	Description
Limit flood	Discards the new MAC addresses.
Limit no-flood	Discards the new MAC addresses. Flooding of unknown unicast packets is disabled.
Shutdown	Disables the bridge domain or bridge port. When the bridge domain is down, none of the bridging functions, such as learning, flooding, forwarding, and so forth take place for the bridge domain. If a bridge port is down as a result of the action, the interface or pseudowire representing the bridge port remains up but the bridge port is not participating in the bridge. When disabled, the port or bridge domain is manually brought up by using an EXEC CLI.

When a limit is exceeded, the system is configured to perform the following notifications:

- Syslog (default)
- Simple Network Management Protocol (SNMP) trap
- Syslog and SNMP trap
- None (no notification)

To clear the MAC limit condition, the number of MACs must go below 75 percent of the configured limit.

MAC Address Withdrawal

For faster VPLS convergence, you can remove or unlearn the MAC addresses that are learned dynamically. The Label Distribution Protocol (LDP) Address Withdrawal message is sent with the list of MAC addresses, which need to be withdrawn to all other PEs that are participating in the corresponding VPLS service.

For the Cisco IOS XR VPLS implementation, a portion of the dynamically learned MAC addresses are cleared by using the MAC addresses aging mechanism by default. The MAC address withdrawal feature is added through the LDP Address Withdrawal message. To enable the MAC address withdrawal feature, use the **withdrawal** command in l2vpn bridge group bridge domain MAC configuration mode. To verify that the MAC address withdrawal is enabled, use the **show l2vpn bridge-domain** command with the **detail** keyword.

Note By default, the LDP MAC Withdrawal feature is disabled on Cisco IOS XR.

The LDP MAC Withdrawal feature is generated due to the following events:

- Attachment circuit goes down. You can remove or add the attachment circuit through the CLI.
- MAC withdrawal messages are received over a VFI pseudowire and are not propagated over access pseudowires. RFC 4762 specifies that both wildcards (by means of an empty Type, Length and Value [TLV]) and a specific MAC address withdrawal. Cisco IOS XR supports only a wildcard MAC address withdrawal.

LSP Ping over VPWS and VPLS

For Cisco IOS XR, the existing support for the Label Switched Path (LSP) ping and traceroute verification mechanisms for point-to-point pseudowires (signaled using LDP FEC128) is extended to cover the pseudowires that are associated with the VFI (VPLS). Currently, the support for the LSP ping and traceroute is limited to manually configured VPLS pseudowires (signaled using LDP FEC128). For information about Virtual Circuit Connection Verification (VCCV) support and the **ping mpls pseudowire** command, see *Cisco IOS XR MPLS Command Reference*.

How to Implement Virtual Private LAN Services

This section describes the tasks that are required to implement VPLS:

- Configuring a Bridge Domain, page MPC-247
- Configuring a Layer 2 Virtual Forwarding Instance, page MPC-257
- Configuring the MAC Address-related Parameters, page MPC-269

Configuring a Bridge Domain

These topics describe how to configure a bridge domain:

- Creating a Bridge Domain, page MPC-247
- Configuring a Pseudowire, page MPC-249
- Associating Members with a Bridge Domain, page MPC-251
- Configuring Bridge Domain Parameters, page MPC-253
- Disabling a Bridge Domain, page MPC-255

Creating a Bridge Domain

Perform this task to create a bridge domain.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

	Command or Action	Purpose

Step 4	`bridge-domain` *bridge-domain name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	`end`
	or
	`commit`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring a Pseudowire

Perform this task to configure a pseudowire under a bridge domain.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **vfi** {*vfi name*}
6. **exit**
7. **neighbor** {*A.B.C.D*} {**pw-id** *value*}
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** RP/0/0/CPU0:router# configure	Enters global configuration mode.
Step 2	**l2vpn** **Example:** RP/0/0/CPU0:router(config)# l2vpn RP/0/0/CPU0:router(config-l2vpn)#	Enters l2vpn configuration mode.
Step 3	**bridge group** *bridge group name* **Example:** RP/0/0/CPU0:router(config-l2vpn)# bridge group csco RP/0/0/CPU0:router(config-l2vpn-bg)#	Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.
Step 4	**bridge-domain** *bridge-domain name* **Example:** RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc RP/0/0/CPU0:router(config-l2vpn-bg-bd)#	Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.
Step 5	**vfi** {*vfi name*} **Example:** RP/0/0/CPU0:router(config-l2vpn-bg-bd)# vfi v1 RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)#	Configures the virtual forwarding interface (VFI) parameters and enters l2vpn bridge group bridge domain VFI configuration mode. • Use the *vfi name* argument to configure the name of the specified virtual forwarding interface.
Step 6	**exit** **Example:** RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# exit RP/0/0/CPU0:router(config-l2vpn-bg-bd)#	Exits the current configuration mode.

	Command or Action	Purpose

Step 7	**neighbor** {*A.B.C.D*} {**pw-id** *value*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# neighbor 10.1.1.2 pw-id 1000`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)#`
		Adds an access pseudowire port to a bridge domain or a pseudowire to a bridge virtual forwarding interface (VFI).
		• Use the *A.B.C.D* argument to specify the IP address of the cross-connect peer.
		• Use the **pw-id** keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.

Step 8	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Associating Members with a Bridge Domain

After a bridge domain is created, perform this task to assign interfaces to the bridge domain. The following types of bridge ports are associated with a bridge domain:

- Ethernet and VLAN
- VFI

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **interface** *interface name*
6. **static-mac-address** {*MAC address*}
7. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**l2vpn**
	
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**bridge group** *bridge group name*
	
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**bridge-domain** *bridge-domain name*
	
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**interface** interface name
	
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/4/0/0`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)#`
		Adds an interface to a bridge domain that allows packets to be forwarded and received from other interfaces that are part of the same bridge domain.

	Command or Action	Purpose

Step 6	**`static-mac-address`** {*`MAC address`*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)# static-mac-address 1.1.1`
		Configures the static MAC address to associate a remote MAC address with a pseudowire or any other bridge interface.

Step 7	**`end`**
	or
	`commit`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Bridge Domain Parameters

To configure the bridge domain parameters, associate the following parameters with a bridge domain:

- Maximum transmission unit (MTU)—Specifies that all members of a bridge domain have the same MTU. The bridge domain member with a different MTU size is not used by the bridge domain even though it is still associated with a bridge domain.
- Flooding—Enables or disables flooding on the bridge domain. By default, flooding is enabled.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **flooding disable**

6. **mtu** *bytes*
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**l2vpn**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**bridge group** *bridge group name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**bridge-domain** *bridge-domain name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**flooding disable** **Example:** `RP/0/0/CPU0:router(config-l2vpn-bg-bd)# flooding disable`	Configures flooding for traffic at the bridge domain level or at the bridge port level.

	Command or Action	Purpose
Step 6	**mtu** *bytes* **Example:** `RP/0/0/CPU0:router(config-l2vpn-bg-bd)# mtu 1000`	Adjusts the maximum packet size or maximum transmission unit (MTU) size for the bridge domain. • Use the *bytes* argument to specify the MTU size, in bytes. The range is from 64 to 65535.

Step 7	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling a Bridge Domain

Perform this task to disable a bridge domain. When a bridge domain is disabled, all VFIs that are associated with the bridge domain are disabled. You are still able to attach or detach members to the bridge domain and the VFIs that are associated with the bridge domain.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **shutdown**
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**l2vpn**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**bridge group** *bridge group name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**bridge-domain** *bridge-domain name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

	Command or Action	Purpose
Step 5	**shutdown** **Example:** `RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`	Shuts down a bridge domain to bring the bridge and all attachment circuits and pseudowires under it to admin down state.

Step 6	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring a Layer 2 Virtual Forwarding Instance

These topics describe how to configure a Layer 2 virtual forwarding instance (VFI):

- Adding the Virtual Forwarding Instance Under the Bridge Domain, page MPC-257
- Associating Pseudowires with the Virtual Forwarding Instance, page MPC-259
- Associating a Virtual Forwarding Instance to a Bridge Domain, page MPC-261
- Attaching Pseudowire Classes to Pseudowires, page MPC-263
- Configuring Any Transport over Multiprotocol Pseudowires By Using Static Labels, page MPC-265
- Disabling a Virtual Forwarding Instance, page MPC-267

Adding the Virtual Forwarding Instance Under the Bridge Domain

Perform this task to create a Layer 2 Virtual Forwarding Instance (VFI) on all provider edge devices under the bridge domain.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **vfi** {*vfi name*}
6. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

	Command or Action	Purpose

Step 5	`vfi {vfi name}`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# vfi v1`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)#`
		Configures virtual forwarding interface (VFI) parameters and enters l2vpn bridge group bridge domain VFI configuration mode.

Step 6	**`end`**
	or
	`commit`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Associating Pseudowires with the Virtual Forwarding Instance

After a VFI is created, perform this task to associate one or more pseudowires with the VFI.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **vfi** {*vfi name*}
6. **neighbor** {*A.B.C.D*} {**pw-id** *value*}
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**`vfi`** `{`*`vfi name`*`}`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# vfi v1`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)#`
		Configures virtual forwarding interface (VFI) parameters and enters l2vpn bridge group bridge domain VFI configuration mode.

	Command or Action	Purpose

Step 6	**neighbor** {*A.B.C.D*} {**pw-id** *value*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# neighbor 10.1.1.2 pw-id 1000`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#`
		Adds an access pseudowire port to a bridge domain or a pseudowire to a bridge virtual forwarding interface (VFI).
		• Use the *A.B.C.D* argument to specify the IP address of the cross-connect peer.
		• Use the **pw-id** keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.

Step 7	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Associating a Virtual Forwarding Instance to a Bridge Domain

Perform this task to associate a VFI to be a member of a bridge domain.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **vfi** {*vfi name*}
6. **neighbor** {*A.B.C.D*} {**pw-id** *value*}

7. **static-mac-address** {*MAC address*}
8. **end**
or
commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**`vfi`** {*`vfi name`*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# vfi v1`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)#`
		Configures virtual forwarding interface (VFI) parameters and enters l2vpn bridge group bridge domain VFI configuration mode.

Step 6	**`neighbor`** {*`A.B.C.D`*} {**`pw-id`** *`value`*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# neighbor 10.1.1.2 pw-id 1000`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#`
		Adds an access pseudowire port to a bridge domain or a pseudowire to a bridge virtual forwarding interface (VFI).
		• Use the *A.B.C.D* argument to specify the IP address of the cross-connect peer.
		• Use the **pw-id** keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.

	Command or Action	Purpose

Step 7	**static-mac-address** {*MAC address*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# static-mac-address 1.1.1`
		Configures the static MAC address to associate a remote MAC address with a pseudowire or any other bridge interface.

Step 8	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Attaching Pseudowire Classes to Pseudowires

Perform this task to attach a pseudowire class to a pseudowire.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **vfi** {*vfi name*}
6. **neighbor** {*A.B.C.D*} {**pw-id** *value*}
7. **pw-class** {*class name*}
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**`vfi`** `{`*`vfi name`*`}`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# vfi v1`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)#`
		Configures virtual forwarding interface (VFI) parameters and enters l2vpn bridge group bridge domain VFI configuration mode.

Step 6	**`neighbor`** `{`*`A.B.C.D`*`} {`**`pw-id`** *`value`*`}`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# neighbor 10.1.1.2 pw-id 1000`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#`
		Adds an access pseudowire port to a bridge domain or a pseudowire to a bridge virtual forwarding interface (VFI).
		• Use the *A.B.C.D* argument to specify the IP address of the cross-connect peer.
		• Use the **pw-id** keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.

	Command or Action	Purpose

Step 7	**pw-class** {*class name*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# pw-class canada`
		Configures the pseudowire class template name to use for the pseudowire.

Step 8	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Any Transport over Multiprotocol Pseudowires By Using Static Labels

Perform this task to configure the Any Transport over Multiprotocol (AToM) pseudowires by using the static labels. A pseudowire becomes a static AToM pseudowire by setting the MPLS static labels to local and remote.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **vfi** {*vfi name*}
6. **neighbor** {*A.B.C.D*} {**pw-id** *value*}

7. **mpls static label** {**local** *value*} {**remote** *value*}
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**`vfi`** `{`*`vfi name`*`}`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# vfi v1`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)#`
		Configures virtual forwarding interface (VFI) parameters and enters l2vpn bridge group bridge domain VFI configuration mode.

Step 6	**`neighbor`** `{`*`A.B.C.D`*`} {`**`pw-id`** *`value`*`}`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# neighbor 10.1.1.2 pw-id 1000`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#`
		Adds an access pseudowire port to a bridge domain or a pseudowire to a bridge virtual forwarding interface (VFI).
		• Use the *A.B.C.D* argument to specify the IP address of the cross-connect peer.
		• Use the **pw-id** keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.

	Command or Action	Purpose

Step 7	**mpls static label** {**local** *value*} {**remote** *value*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# mpls static label local 800 remote 500`
		Configures the MPLS static labels and the static labels for the access pseudowire configuration. You can set the local and remote pseudowire labels.

Step 8	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling a Virtual Forwarding Instance

Perform this task to disable a VFI. When a VFI is disabled, all the previously established pseudowires that are associated with the VFI are disconnected. LDP advertisements are sent to withdraw the MAC addresses that are associated with the VFI. However, you can still attach or detach attachment circuits with a VFI after a shutdown.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **vfi** {*vfi name*}
6. **shutdown**

7. **end**
   or
   commit
8. **show l2vpn bridge-domain** [**detail**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**`vfi`** `{`*`vfi name`*`}`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# vfi v1`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)#`
		Configures virtual forwarding interface (VFI) parameters and enters l2vpn bridge group bridge domain VFI configuration mode.

Step 6	**`shutdown`**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# shutdown`
		Disables the virtual forwarding interface (VFI).

	Command or Action	Purpose

Step 7	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-vfi)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 8	**`show l2vpn bridge-domain`** `[`**`detail`**`]`
	
	`Example:`
	`RP/0/0/CPU0:router# show l2vpn bridge-domain detail`
		Displays the state of the VFI. For example, if you shut down the VFI, the VFI is shown as shut down under the bridge domain.

Configuring the MAC Address-related Parameters

These topics describe how to configure the MAC address-related parameters:

- Configuring the MAC Address Source-based Learning, page MPC-269
- Enabling the MAC Address Withdrawal, page MPC-271
- Configuring the MAC Address Limit, page MPC-274
- Configuring the MAC Address Aging, page MPC-276

The MAC table attributes are set for the bridge domains.

Configuring the MAC Address Source-based Learning

Perform this task to configure the MAC address source-based learning.

SUMMARY STEPS

1. **configure**
2. **l2vpn**

3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **mac**
6. **learning disable**
7. **end**
or
commit
8. **show l2vpn bridge-domain** [**detail**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**`mac`**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# mac`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)#`
		Enters l2vpn bridge group bridge domain MAC configuration mode.

Step 6	**`learning disable`**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# learning disable`
		Overrides the MAC learning configuration of a parent bridge or sets the MAC learning configuration of a bridge.

	Command or Action	Purpose

Step 7	`end`
	or
	`commit`
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 8	**`show l2vpn bridge-domain`** `[`**`detail`**`]`
	Example:
	`RP/0/0/CPU0:router# show l2vpn bridge-domain detail`
		Displays the details that the MAC address source-based learning is disabled on the bridge.

Enabling the MAC Address Withdrawal

Perform this task to enable the MAC address withdrawal for a specified bridge domain.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **mac**
6. **withdrawal**
7. **end**
   or
   commit
8. **show l2vpn bridge-domain** [**detail**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure`**
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

Step 2	**`l2vpn`**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**`bridge group`** *`bridge group name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**`bridge-domain`** *`bridge-domain name`*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**`mac`**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# mac`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)#`
		Enters l2vpn bridge group bridge domain MAC configuration mode.

Step 6	**`withdrawal`**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# withdrawal`
		Enables the MAC address withdrawal for a specified bridge domain.

	Command or Action	Purpose

Step 7	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 8	**show l2vpn bridge-domain** [**detail**] **Example:** `P/0/RP0/CPU0:router# show l2vpn bridge-domain detail`	Displays detailed sample output to specify that the MAC address withdrawal is enabled. In addition, the sample output displays the number of MAC withdrawal messages that are sent over or received from the pseudowire.

The following sample output shows the MAC address withdrawal fields:

```
RP/0/0/CPU0:router# show l2vpn bridge-domain detail

Bridge group: siva_group, bridge-domain: siva_bd, id: 0, state: up, ShgId: 0, MSTi: 0
  MAC Learning: enabled
  MAC withdraw: enabled
  Flooding:
    Broadcast & Multicast: enabled
    Unknown Unicast: enabled
  MAC address aging time: 300 s Type: inactivity
  MAC address limit: 4000, Action: none, Notification: syslog
  MAC limit reached: no
  Security: disabled
  DHCPv4 Snooping: disabled
  MTU: 1500
  MAC Filter:  Static MAC addresses:
  ACs: 1 (1 up), VFIs: 1, PWs: 2 (1 up)
  List of ACs:
    AC: GigabitEthernet0/4/0/1, state is up
      Type Ethernet
      MTU 1500; XC ID 0x5000001; interworking none; MSTi 0 (unprotected)
      MAC Learning: enabled
      MAC withdraw: disabled
    Flooding:
      Broadcast & Multicast: enabled
      Unknown Unicast: enabled
    MAC address aging time: 300 s Type: inactivity
    MAC address limit: 4000, Action: none, Notification: syslog
    MAC limit reached: no
    Security: disabled
    DHCPv4 Snooping: disabled
    Static MAC addresses:
    Statistics:
      packet totals: receive 6,send 0
      byte totals: receive 360,send 4
List of Access PWs:
List of VFIs:
  VFI siva_vfi
    PW: neighbor 1.1.1.1, PW ID 1, state is down ( local ready )
      PW class not set, XC ID 0xff000001
      Encapsulation MPLS, protocol LDP
      PW type Ethernet, control word enabled, interworking none
      PW backup disable delay 0 sec
      Sequencing not set
              MPLS         Local                          Remote
        ------------ ------------------------------ -------------------------
             Label          30005                          unknown
             Group ID     0x0                             0x0
             Interface    siva/vfi                         unknown
             MTU          1500                            unknown
             Control word enabled                    unknown
             PW type      Ethernet                  unknown
        ------------ ------------------------------ -------------------------
      Create time: 19/11/2007 15:20:14 (00:25:25 ago)
      Last time status changed: 19/11/2007 15:44:00 (00:01:39 ago)
      MAC withdraw message: send 0 receive 0
```

Configuring the MAC Address Limit

Perform this task to configure the parameters for the MAC address limit.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **mac**
6. **limit**
7. **maximum** {*value*}
8. **action** {**flood** | **no-flood** | **shutdown**}
9. **notification** {**both** | **none** | **trap**}
10. **end**
    or
    commit
11. **show l2vpn bridge-domain** [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**l2vpn**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**bridge group** *bridge group name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**bridge-domain** *bridge-domain name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**mac**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# mac`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)#`
		Enters l2vpn bridge group bridge domain MAC configuration mode.

Step 6	**limit**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# limit`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-limit)#`
		Sets the MAC address limit for action, maximum, and notification and enters l2vpn bridge group bridge domain MAC limit configuration mode.

Step 7	**maximum** {*value*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-limit)# maximum 5000`
		Configures the specified action when the number of MAC addresses learned on a bridge is reached.

Step 8	**action** {**flood** | **no-flood** | **shutdown**}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-limit)# action flood`
		Configures the bridge behavior when the number of learned MAC addresses exceed the MAC limit configured.

	Command or Action	Purpose

Step 9	**notification** {**both** | **none** | **trap**}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-limit)# notification both`
		Specifies the type of notification that is sent when the number of learned MAC addresses exceeds the configured limit.

Step 10	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-limit)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-limit)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 11	**show l2vpn bridge-domain** [**detail**] **Example:** `RP/0/0/CPU0:router# show l2vpn bridge-domain detail`	Displays the details about the MAC address limit.

Configuring the MAC Address Aging

Perform this task to configure the parameters for MAC address aging.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge group name*
4. **bridge-domain** *bridge-domain name*
5. **mac**
6. **aging**
7. **time** {*seconds*}

8. **type** {**absolute** | **inactivity**}
9. **end**
   or
   commit
10. **show l2vpn bridge-domain** [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**l2vpn**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters l2vpn configuration mode.

Step 3	**bridge group** *bridge group name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Step 4	**bridge-domain** *bridge-domain name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.

Step 5	**mac**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# mac`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)#`
		Enters l2vpn bridge group bridge domain MAC configuration mode.

Step 6	**aging**
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac)# aging`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-aging)#`
		Enters the MAC aging configuration submode to set the aging parameters such as time and type.

Step 7	**time** {*seconds*}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-aging)# time 300`
		Configures the maximum aging time.
		• Use the *seconds* argument to specify the maximum age of the MAC address table entry. The range is from 120 to 1000000 seconds. Aging time is counted from the last time that the switch saw the MAC address. The default value is 300 seconds.

	Command or Action	Purpose

Step 8	**type** {**absolute** | **inactivity**}
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-aging)# type absolute`
		Configures the type for MAC address aging.
		• Use the **absolute** keyword to configure the absolute aging type.
		• Use the **inactivity** keyword to configure the inactivity aging type.

Step 9	**end**
	or
	commit
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-aging)# end`
	or
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-mac-aging)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 10	**show l2vpn bridge-domain** [**detail**] **Example:** `RP/0/0/CPU0:router# show l2vpn bridge-domain detail`	Displays the details about the aging fields.

Configuration Examples for Virtual Private LAN Services

This section includes the following configuration examples:

- Virtual Private LAN Services Configuration for Provider Edge-to-Provider Edge: Example, page MPC-279
- Virtual Private LAN Services Configuration for Provider Edge-to-Customer Edge: Example, page MPC-280

Virtual Private LAN Services Configuration for Provider Edge-to-Provider Edge: Example

These configuration examples show how to create a Layer 2 VFI with a full-mesh of participating VPLS provider edge (PE) nodes.

The following configuration example shows how to configure PE 1:

```
configure
 l2vpn
  bridge group 1
   bridge-domain PE1-VPLS-A
    GigabitEthernet0/0---AC
     exit
    vfi 1
     neighbor 2.2.2.2 pw-id 1---PW1
     neighbor 3.3.3.3 pw-id 1---PW2
     !
   !
 interface loopback 0
  ipv4 address 1.1.1.1 255.255.255.25
  commit
```

The following configuration example shows how to configure PE 2:

```
configure
 l2vpn
  bridge group 1
   bridge-domain PE2-VPLS-A
    interface GigabitEthernet0/0---AC
     exit
    vfi 1
     neighbor 1.1.1.1 pw-id 1---PW1
     neighbor 3.3.3.3 pw-id 1---PW2
     !
   !
 interface loopback 0
  ipv4 address 2.2.2.2 255.255.255.25
  commit
```

The following configuration example shows how to configure PE 3:

```
configure
 l2vpn
  bridge group 1
   bridge-domain PE3-VPLS-A
    interface GigabitEthernet0/0---AC
     exit
    vfi 1
     neighbor 1.1.1.1 pw-id 1---PW1
     neighbor 2.2.2.2 pw-id 1---PW2
     !
   !
 interface loopback 0
  ipv4 address 3.3.3.3 255.255.255.25
  commit
```

Virtual Private LAN Services Configuration for Provider Edge-to-Customer Edge: Example

The following configuration shows how to configure VPLS for a PE-to-CE nodes:

```
configure
 interface GigabitEthernet0/0
  l2transport---AC interface
   exit
  no ipv4 address
  no ipv4 directed-broadcast
  negotiation auto
  no cdp enable
  end

configure
 interface GigabitEthernet0/0
  l2transport
   exit
  no ipv4 address
  no ipv4 directed-broadcast
  negotiation auto
  no cdp enable
  end

configure
 interface GigabitEthernet0/0
  l2transport
   exit
  no ipv4 address
  no ipv4 directed-broadcast
  negotiation auto
  no cdp enable
```

Additional References

For additional information related to implementing VPLS, refer to the following references:

Related Documents

Related Topic	Document Title
Cisco IOS XR L2VPN command reference document	*MPLS Virtual Private Network Commands on Cisco IOS XR Software* module in *Cisco IOS XR MPLS Command Reference*
MPLS VPLS-related commands	*MPLS Virtual Private LAN Services Commands on Cisco IOS XR Software* module in *Cisco IOS XR MPLS Command Reference*
MPLS Layer 2 VPNs	*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software* module in *Cisco IOS XR MPLS Configuration Guide*
MPLS VPNs over IP Tunnels	*MPLS VPNs over IP Tunnels on Cisco IOS XR Software* module in *Cisco IOS XR MPLS Configuration Guide*

Related Topic	**Document Title**
Cisco CRS-1 router getting started material	*Cisco IOS XR Getting Started Guide*
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*

Standards

Standards[1]	**Title**
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

[1] Not all supported standards are listed.

MIBs

MIBs	**MIBs Link**
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	**Title**
RFC 3931	*Layer Two Tunneling Protocol - Version 3 (L2TPv3)*
RFC 4447	*Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)*, April 2006
RFC 4448	*Encapsulation Methods for Transport of Ethernet over MPLS Networks*, April 2006

Technical Assistance

Description	**Link**
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport