Implementing Multipoint Layer 2 Services

This module provides the conceptual and configuration information for Multipoint Layer 2 Bridging Services, also called Virtual Private LAN Services (VPLS) on Cisco ASR 9000 Series Aggregation Services Routers. VPLS supports Layer 2 VPN technology and provides transparent multipoint Layer 2 connectivity for customers.

Note

This approach enables service providers to host a multitude of new services such as broadcast TV and Layer 2 VPNs. For more information about MPLS Layer 2 VPN on Cisco ASR 9000 Series Routers 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 Multipoint Layer 2 Services on Cisco ASR 9000 Series Routers

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 3.7.2</td>
<td>This feature was introduced on Cisco ASR 9000 Series Routers.</td>
</tr>
<tr>
<td>Release 3.9.0</td>
<td>These features were added:</td>
</tr>
<tr>
<td></td>
<td>• Blocking unknown unicast flooding.</td>
</tr>
<tr>
<td></td>
<td>• Disabling MAC flush.</td>
</tr>
<tr>
<td></td>
<td>• Multiple Spanning Tree Access Gateway</td>
</tr>
<tr>
<td></td>
<td>• Scale enhancements were introduced. See Table 1 on page 497 for more information on scale enhancements.</td>
</tr>
<tr>
<td>Release 3.9.1</td>
<td>Support for VPLS with BGP Autodiscovery and LDP Signaling was added.</td>
</tr>
<tr>
<td>Release 4.0.1</td>
<td>Support was added for the following features:</td>
</tr>
<tr>
<td></td>
<td>• Dynamic ARP Inspection</td>
</tr>
<tr>
<td></td>
<td>• IP SourceGuard</td>
</tr>
<tr>
<td></td>
<td>• MAC Address Security</td>
</tr>
<tr>
<td>Release 4.1.0</td>
<td>Support was added for these VPLS features on the ASR 9000 SIP-700 line card:</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• MAC learning and forwarding</td>
</tr>
<tr>
<td></td>
<td>• MAC address aging support</td>
</tr>
<tr>
<td></td>
<td>• MAC Limiting</td>
</tr>
<tr>
<td></td>
<td>• Split Horizon Group</td>
</tr>
<tr>
<td></td>
<td>• MAC address Withdrawal</td>
</tr>
<tr>
<td></td>
<td>• Flooding of unknown unicast, broadcast and multicast packets</td>
</tr>
<tr>
<td></td>
<td>• Access pseudowire</td>
</tr>
<tr>
<td></td>
<td>• H-VPLS PW-access</td>
</tr>
<tr>
<td></td>
<td>• PW redundancy</td>
</tr>
</tbody>
</table>

Support was added for the G.8032 Ethernet Ring Protection feature.

<table>
<thead>
<tr>
<th>Release 4.2.1</th>
<th>Support was added for Flow Aware Transport (FAT) Pseudowire feature.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Release 4.3.0</th>
<th>Support was added for these features:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Pseudowire Headend (PWHE)</td>
</tr>
<tr>
<td></td>
<td>• Scale enhancements on ASR 9000 Enhanced Ethernet line card:</td>
</tr>
<tr>
<td></td>
<td>– Support for 128000 pseudowires within VPWS and VPLS</td>
</tr>
<tr>
<td></td>
<td>– Support for 128000 pseudowires across VPLS and VPWS instances</td>
</tr>
<tr>
<td></td>
<td>– Support for upto 512 pseudowires in a bridge</td>
</tr>
<tr>
<td></td>
<td>– Support for 128000 bundle attachment circuits</td>
</tr>
<tr>
<td></td>
<td>– Support for 128000 VLANs</td>
</tr>
<tr>
<td></td>
<td>• L2VPN over GRE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Release 4.3.1</th>
<th>Support was added for:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• VC type 4 in VPLS with BGP Autodiscovery</td>
</tr>
<tr>
<td></td>
<td>• IPv6 support for PWHE</td>
</tr>
</tbody>
</table>
Prerequisites for Implementing Multipoint Layer 2 Services

Before configuring VPLS, ensure that these tasks and conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command.
  
  If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

- Configure IP routing in the core so that the provider edge (PE) routers can reach each other through IP.

- 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.

- Configure MPLS and Label Distribution Protocol (LDP) in the core so that a label switched path (LSP) exists between the PE routers.

Information About Implementing Multipoint Layer 2 Services

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

- Virtual Private LAN Services Overview, page 236
- VPLS for an MPLS-based Provider Core, page 239
- VPLS Discovery and Signaling, page 240
- MAC Address-related Parameters, page 243
- LSP Ping over VPWS and VPLS, page 246
- Split Horizon Groups, page 247
- Layer 2 Security, page 247
- G.8032 Ethernet Ring Protection, page 248
- Flow Aware Transport Pseudowire (FAT PW), page 253
- Pseudowire Headend, page 254
- L2VPN over GRE, page 254
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.

Some of the components present in a VPLS network are described in these sections.

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 for pseudowires under the same VFI. However, in the default configuration, split horizon is not enabled on the attachment circuits (interfaces or pseudowires).

Flood Optimization

A Cisco ASR 9000 Series Router, while bridging traffic in a bridge domain, minimizes the amount of traffic that floods unnecessarily. The Flood Optimization feature accomplishes this functionality. However, in certain failure recovery scenarios, extra flooding is actually desirable in order to prevent traffic loss. Traffic loss occurs during a temporary interval when one of the bridge port links becomes inactive, and a standby link replaces it.

In some configurations, optimizations to minimize traffic flooding is achieved at the expense of traffic loss during the short interval in which one of the bridge's links fails, and a standby link replaces it. Therefore, Flood Optimization can be configured in different modes to specify a particular flooding behavior suitable for your configuration.

These flood optimization modes can be configured:

- Bandwidth Optimization Mode
- Convergence Mode
- TE FRR Optimized Mode

Bandwidth Optimization Mode

Flooded traffic is sent only to the line cards on which a bridge port or pseudowire that is attached to the bridge domain resides. This is the default mode.

Convergence Mode

Flooded traffic is sent to all line cards in the system. Traffic is flooded regardless of whether they have a bridge port or a pseudowire that is attached to the bridge domain. If there are multiple Equal Cost MPLS Paths (ECMPs) attached to that bridge domain, traffic is flooded to all ECMPs.

The purpose of Convergence Mode is to ensure that an absolute minimum amount of traffic is lost during the short interval of a bridge link change due to a failure.
TE FRR Optimized Mode

The Traffic Engineering Fast Reroute (TE FRR) Optimized Mode is similar to the Bandwidth Optimized Mode, except for the flooding behavior with respect to any TE FRR pseudowires attached to the bridge domain. In TE FRR Optimized Mode, traffic is flooded to both the primary and backup FRR interfaces. This mode is used to minimize traffic loss during an FRR failover, thus ensuring that the bridge traffic complies with the FRR recovery time constraints.

Dynamic ARP Inspection

Dynamic ARP Inspection (DAI) is a method of providing protection against address resolution protocol (ARP) spoofing attacks. It intercepts, logs, and discards ARP packets with invalid IP-to-MAC address bindings. This capability protects the network from certain man-in-the-middle attacks. The DAI feature is disabled by default.

ARP enables IP communication within a Layer 2 broadcast domain by mapping an IP address to a MAC address. Spoofing attacks occur because ARP allows a response from a host even when an ARP request is not actually received. After an attack occurs, all traffic, from the device under attack, first flows through the attacker's system, and then to the router, switch, or the host. An ARP spoofing attack affects the devices connected to your Layer 2 network by sending false information to the ARP caches of the devices connected to the subnet. The sending of false information to an ARP cache is known as ARP cache poisoning.

The Dynamic ARP Inspection feature ensures that only valid ARP requests and responses are relayed. There are two types of ARP inspection:

- Mandatory inspection—The sender’s MAC address, IPv4 address, receiving bridge port XID and bridge are checked.
- Optional inspection—The following items are validated:
  - Source MAC: The sender’s and source MACs are checked. The check is performed on all ARP or RARP packets.
  - Destination MAC: The target and destination MACs are checked. The check is performed on all Reply or Reply Reverse packets.
  - IPv4 Address: For ARP requests, a check is performed to verify if the sender’s IPv4 address is 0.0.0.0, a multicast address or a broadcast address. For ARP Reply and ARP Reply Reverse, a check is performed to verify if the target IPv4 address is 0.0.0.0, a multicast address or a broadcast address. This check is performed on Request, Reply and Reply Reverse packets.

Note

The DAI feature is supported on attachment circuits and EFPs. Currently, the DAI feature is not supported on pseudowires.

IP Source Guard

IP source guard (IPSG) is a security feature that filters traffic based on the DHCP snooping binding database and on manually configured IP source bindings in order to restrict IP traffic on non-routed Layer 2 interfaces.

The IPSG feature provides source IP address filtering on a Layer 2 port, to prevent a malicious hosts from manipulating a legitimate host by assuming the legitimate host's IP address. This feature uses dynamic DHCP snooping and static IP source binding to match IP addresses to hosts.
Initially, all IP traffic, except for DHCP packets, on the EFP configured for IPSG is blocked. After a client receives an IP address from the DHCP server, or after static IP source binding is configured by the administrator, all traffic with that IP source address is permitted from that client. Traffic from other hosts is denied. This filtering limits a host's ability to attack the network by claiming a neighbor host's IP address.

Note

The IPSG feature is supported on attachment circuits and EFPs. Currently, the IPSG feature is not supported on pseudowires.

Pseudowires

A pseudowire is a point-to-point connection between pairs of PE routers. Its primary function is to emulate services like Ethernet over an underlying core MPLS network through encapsulation into a common MPLS format. By encapsulating services into a common MPLS format, a pseudowire allows carriers to converge their services to an MPLS network.

The following scale enhancements are applicable to ASR 9000 Enhanced Ethernet line card:

- Support for 128000 pseudowires within VPWS and VPLS
- Support for 128000 pseudowires across VPLS and VPWS instances
- Support for up to 512 pseudowires in a bridge

Note

This scale enhancement is supported in hardware configurations where RSP3 and ASR 9000 Enhanced Ethernet line cards are used. However, these enhancements are not applicable to the RSP2, ASR 9000 Ethernet Line Card and Cisco ASR 9000 Series SPA Interface Processor-700 line cards.

DHCP Snooping over Pseudowire

The Cisco ASR 9000 Series Routers provide the ability to perform DHCP snooping, where the DHCP server is reachable on a pseudowire. The Pseudowire is considered as a trusted interface.

The `dhcp ipv4 snoop profile {dhcp-snooping-profile1}` command is provided under the bridge domain to enable DHCP snooping on a bridge and to attach a DHCP snooping profile to the bridge.

Virtual Forwarding Instance

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.

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.
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. A bridge domain, which is the building block for multipoint bridging, is present 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.

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.

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 virtual circuits (VCs) among them.

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 Architecture

The basic or flat VPLS architecture allows for the end-to-end connection between the provider edge (PE) routers to provide multipoint ethernet services. Figure 1 shows a flat VPLS architecture illustrating the interconnection between the network provider edge (N-PE) nodes over an IP/MPLS network.

Figure 1 Basic VPLS Architecture

The VPLS network requires the creation of a bridge domain (Layer 2 broadcast domain) on each of the PE routers. 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.
The PEs in the VPLS architecture are connected with a full mesh of Pseudowires (PWs). A Virtual Forwarding Instance (VFI) is used to interconnect the mesh of pseudowires. A bridge domain is connected to a VFI to create a Virtual Switching Instance (VSI), that provides Ethernet multipoint bridging over a PW mesh. VPLS network links the VSIs using the MPLS pseudowires to create an emulated Ethernet Switch.

With VPLS, all customer equipment (CE) devices participating in a single VPLS instance appear to be on the same LAN and, therefore, can communicate directly with one another in a multipoint topology, without requiring a full mesh of point-to-point circuits at the CE device. 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 IEEE 802.3, VLAN IEEE 802.1aq, and VLAN-in-VLAN (q-in-q) traffic across multiple sites that belong to the same Layer 2 broadcast domain. VPLS offers simple VLAN 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 PE routers. The VPLS implementation is based on Label Distribution Protocol (LDP)-based pseudowire signaling.

VPLS for Layer 2 Switching

VPLS technology includes the capability of configuring the Cisco ASR 9000 Series Routers to perform Layer 2 bridging. In this mode, the Cisco ASR 9000 Series Routers can be configured to operate like other Cisco switches.

These features are supported:
- Bridging IOS XR Trunk Interfaces
- Bridging on EFPs

Refer to the Configuration Examples for Multipoint Layer 2 Services section for examples on these bridging features.

VPLS Discovery and Signaling

VPLS is a Layer 2 multipoint service and it emulates LAN service across a WAN service. VPLS enables service providers to interconnect several LAN segments over a packet-switched network and make it behave as one single LAN. Service provider can provide a native Ethernet access connection to customers using VPLS.

The VPLS control plane consists of two important components, autodiscovery and signaling:
- VPLS Autodiscovery eliminates the need to manually provision VPLS neighbors. VPLS Autodiscovery enables each VPLS PE router to discover the other provider edge (PE) routers that are part of the same VPLS domain.
- Once the PEs are discovered, pseudowires (PWs) are signaled and established across each pair of PE routers forming a full mesh of PWs across PE routers in a VPLS domain.
BGP-based VPLS Autodiscovery

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. Autodiscovery requires this information to be distributed to all members of a VPN. VPLS is a multipoint mechanism for which BGP is well suited.

BGP-based VPLS autodiscovery eliminates the need to manually provision VPLS neighbors. VPLS autodiscovery enables each VPLS PE router to discover the other provider edge (PE) routers that are part of the same VPLS domain. VPLS Autodiscovery also tracks when PE routers are added to or removed from the VPLS domain. When the discovery process is complete, each PE router has the information required to setup VPLS pseudowires (PWs).

BGP Auto Discovery With BGP Signaling

The implementation of VPLS in a network requires the establishment of a full mesh of PWs between the provider edge (PE) routers. The PWs can be signaled using BGP signaling.
The BGP Network Layer Reachability Information (NLRI) takes care of the above two components simultaneously. The NLRI generated by a given PE contains the necessary information required by any other PE. These components enable the automatic setting up of a full mesh of pseudowires for each VPLS without having to manually configure those pseudowires on each PE.

**NLRI Format for VPLS with BGP AD and Signaling**

Figure 4 shows the NLRI format for VPLS with BGP AD and Signaling.

**Figure 4  NLRI Format**

<table>
<thead>
<tr>
<th>Length (2 octets)</th>
<th>Route Distinguisher (8 octets)</th>
<th>VE ID (2 octets)</th>
<th>VE Block Offset (2 octets)</th>
<th>VE Block Size (2 octets)</th>
<th>Label Base (3 octets)</th>
</tr>
</thead>
</table>

**BGP Auto Discovery With LDP Signaling**

Signaling of pseudowires requires exchange of information between two endpoints. Label Distribution Protocol (LDP) is better suited for point-to-point signaling. The signaling of pseudowires between provider edge devices, uses targeted LDP sessions to exchange label values and attributes and to configure the pseudowires.

**Figure 5  Discovery and Signaling Attributes**

A PE router advertises an identifier through BGP for each VPLS. This identifier is unique within the VPLS instance and acts like a VPLS ID. The identifier enables the PE router receiving the BGP advertisement to identify the VPLS associated with the advertisement and import it to the correct VPLS instance. In this manner, for each VPLS, a PE router learns the other PE routers that are members of the VPLS.
The LDP protocol is used to configure a pseudowire to all the other PE routers. FEC 129 is used for the signaling. The information carried by FEC 129 includes the VPLS ID, the Target Attachment Individual Identifier (TAII) and the Source Attachment Individual Identifier (SAII).

The LDP advertisement also contains the inner label or VPLS label that is expected for the incoming traffic over the pseudowire. This enables the LDP peer to identify the VPLS instance with which the pseudowire is to be associated and the label value that it is expected to use when sending traffic on that pseudowire.

**NLRI and Extended Communities**

Figure 6 depicts Network Layer Reachability Information (NLRI) and extended communities (Ext Comms).

**Figure 6 NLRI and Extended Communities**

<table>
<thead>
<tr>
<th>NLRI:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length (2 octets)</td>
</tr>
<tr>
<td>Route Distinguisher (8 octets)</td>
</tr>
<tr>
<td>L2VPN Router ID (4 octets)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ext Comms:</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPLS-ID (8 octets)</td>
</tr>
<tr>
<td>Route Target (8 octets)</td>
</tr>
</tbody>
</table>

**Interoperability Between Cisco IOS XR and Cisco IOS on VPLS LDP Signaling**

The Cisco IOS Software encodes the NLRI length in the first byte in bits format in the BGP Update message. However, the Cisco IOS XR Software interprets the NLRI length in 2 bytes. Therefore, when the BGP neighbor with VPLS-VPWS address family is configured between the IOS and the IOS XR, NLRI mismatch can happen, leading to flapping between neighbors. To avoid this conflict, IOS supports `prefix-length-size 2` command that needs to be enabled for IOS to work with IOS XR. When the `prefix-length-size 2` command is configured in IOS, the NLRI length is encoded in bytes. This configuration is mandatory for IOS to work with IOS XR.

This is a sample IOS configuration with the `prefix-length-size 2` command:

```
router bgp 1
    address-family l2vpn vpls
    neighbor 5.5.5.2 activate
    neighbor 5.5.5.2 prefix-length-size 2 ---------> NLRI length = 2 bytes
    exit-address-family
```

**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.

These topics provide information about the MAC address-related parameters:
Information About Implementing Multipoint Layer 2 Services

- MAC Address Flooding, page 244
- MAC Address-based Forwarding, page 244
- MAC Address Source-based Learning, page 244
- MAC Address Aging, page 245
- MAC Address Limit, page 245
- MAC Address Withdrawal, page 246
- MAC Address Security, page 246

**Note**

After you modify the MAC limit or action at the bridge domain level, ensure that you shut and unshut the bridge domain for the action to take effect. If you modify the MAC limit or action on an attachment circuit (through which traffic is passing), the attachment circuit must be shut and unshut for the action to take effect.

**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 pseudowire, 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 at the port level. The bridge domain level limit is always configured and cannot be disabled. The default value of the bridge domain level limit is 4000 and can be changed in the range of 1-512000.

Note

Configuring MAC address limit at the port level is only supported on the ASR 9000 Enhanced Ethernet Line Card.

When the MAC address limit is violated, the system is configured to take one of the actions that are listed in Table 1.

Table 1  MAC Address Limit Actions

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit flood</td>
<td>Discards the new MAC addresses.</td>
</tr>
<tr>
<td>Limit no-flood</td>
<td>Discards the new MAC addresses. Flooding of unknown unicast packets is disabled.</td>
</tr>
<tr>
<td>Limit shutdown</td>
<td>Disables forwarding MAC addresses.</td>
</tr>
</tbody>
</table>

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

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

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

The LDP MAC Withdrawal feature is generated due to these 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 software supports only a wildcard MAC address withdrawal.

MAC Address Security

You can configure MAC address security at the interfaces and at the bridge access ports (subinterfaces) levels. However, MAC security configured under an interface takes precedence to MAC security configured at the bridge domain level. When a MAC address is first learned, on an EFP that is configured with MAC security and then, the same MAC address is learned on another EFP, these events occur:
- the packet is dropped
- the second EFP is shutdown
- the packet is learned and the MAC from the original EFP is flushed

LSP Ping over VPWS and VPLS

For Cisco IOS XR software, 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 the Cisco ASR 9000 Series Aggregation Services Router MPLS Command Reference.
Split Horizon Groups

An IOS XR bridge domain aggregates attachment circuits (ACs) and pseudowires (PWs) in one of three groups called Split Horizon Groups. When applied to bridge domains, Split Horizon refers to the flooding and forwarding behavior between members of a Split Horizon group. In general, frames received on one member of a split horizon group are not flooded out to the other members of the same group.

Bridge Domain traffic is either unicast or multicast.

Flooding traffic consists of unknown unicast destination MAC address frames; frames sent to Ethernet multicast addresses (Spanning Tree BPDUs, etc.); Ethernet broadcast frames (MAC address FF-FF-FF-FF-FF-FF).

Known Unicast traffic consists of frames sent to bridge ports that were learned from that port using MAC learning.

Traffic flooding is performed for broadcast, multicast and unknown unicast destination address.

<table>
<thead>
<tr>
<th>Split Horizon Group</th>
<th>Who belongs to this Group?</th>
<th>Multicast within Group</th>
<th>Unicast within Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Default—any member not covered by groups 1 or 2.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>Any PW configured under VFI.</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Any AC or PW configured with <code>split-horizon</code> keyword.</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Important notes on Split Horizon Groups:

- All bridge ports or PWs that are members of a bridge domain must belong to one of the three groups.
- By default, all bridge ports or PWs are members of group 0.
- The VFI configuration submode under a bridge domain configuration indicates that members under this domain are included in group 1.
- A PW that is configured in group 0 is called an Access Pseudowire.
- The `split-horizon group` command is used to designate bridge ports or PWs as members of group 2.
- The ASR9000 only supports one VFI group.

Layer 2 Security

These topics describe the Layer 2 VPN extensions to support Layer 2 security:

- Port Security, page 248
- Dynamic Host Configuration Protocol Snooping, page 248
Port Security

Use port security with dynamically learned and static MAC addresses to restrict a port’s ingress traffic by limiting the MAC addresses that are allowed to send traffic into the port. When secure MAC addresses are assigned to a secure port, the port does not forward ingress traffic that has source addresses outside the group of defined addresses. If the number of secure MAC addresses is limited to one and assigned a single secure MAC address, the device attached to that port has the full bandwidth of the port.

These port security features are supported:
- Limits the MAC table size on a bridge or a port.
- Facilitates actions and notifications for a MAC address.
- Enables the MAC aging time and mode for a bridge or a port.
- Filters static MAC addresses on a bridge or a port.
- Marks ports as either secure or nonsecure.
- Enables or disables flooding on a bridge or a port.

After you have set the maximum number of secure MAC addresses on a port, you can configure port security to include the secure addresses in the address table in one of these ways:
- Statically configure all secure MAC addresses by using the `static-address` command.
- Allow the port to dynamically configure secure MAC addresses with the MAC addresses of connected devices.
- Statically configure a number of addresses and allow the rest to be dynamically configured.

Dynamic Host Configuration Protocol Snooping

Dynamic Host Configuration Protocol (DHCP) snooping is a security feature that acts like a firewall between untrusted hosts and trusted DHCP servers. The DHCP snooping feature performs these activities:
- Validates DHCP messages received from untrusted sources and filters out invalid messages.
- Rate-limits DHCP traffic from trusted and untrusted sources.
- Builds and maintains the binding database of DHCP snooping, which contains information about untrusted hosts with leased IP addresses.
- Utilizes the binding database of DHCP snooping to validate subsequent requests from untrusted hosts.

For additional information regarding DHCP, see the *Cisco ASR 9000 Series Aggregation Services Router IP Addresses and Services Configuration Guide*.

G.8032 Ethernet Ring Protection

Ethernet Ring Protection (ERP) protocol, defined in ITU-T G.8032, provides protection for Ethernet traffic in a ring topology, while ensuring that there are no loops within the ring at the Ethernet layer. The loops are prevented by blocking either a pre-determined link or a failed link.
Overview

Each Ethernet ring node is connected to adjacent Ethernet ring nodes participating in the Ethernet ring using two independent links. A ring link never allows formation of loops that affect the network. The Ethernet ring uses a specific link to protect the entire Ethernet ring. This specific link is called the ring protection link (RPL). A ring link is bound by two adjacent Ethernet ring nodes and a port for a ring link (also known as a ring port).

The minimum number of Ethernet ring nodes in an Ethernet ring is two.

The fundamentals of ring protection switching are:

- the principle of loop avoidance
- the utilization of learning, forwarding, and Filtering Database (FDB) mechanisms

Loop avoidance in an Ethernet ring is achieved by ensuring that, at any time, traffic flows on all but one of the ring links which is the RPL. Multiple nodes are used to form a ring:

- RPL owner—It is responsible for blocking traffic over the RPL so that no loops are formed in the Ethernet traffic. There can be only one RPL owner in a ring.
- RPL neighbor node—The RPL neighbor node is an Ethernet ring node adjacent to the RPL. It is responsible for blocking its end of the RPL under normal conditions. This node type is optional and prevents RPL usage when protected.
- RPL next-neighbor node—The RPL next-neighbor node is an Ethernet ring node adjacent to RPL owner node or RPL neighbor node. It is mainly used for FDB flush optimization on the ring. This node is also optional.

Figure 7 illustrates the G.8032 Ethernet ring.

Figure 7 G.8032 Ethernet Ring

Nodes on the ring use control messages called RAPS to coordinate the activities of switching on or off the RPL link. Any failure along the ring triggers a RAPS signal fail (RAPS SF) message along both directions, from the nodes adjacent to the failed link, after the nodes have blocked the port facing the failed link. On obtaining this message, the RPL owner unblocks the RPL port.

A single link failure in the ring ensures a loop-free topology.
Line status and Connectivity Fault Management protocols are used to detect ring link and node failure. During the recovery phase, when the failed link is restored, the nodes adjacent to the restored link send RAPS no request (RAPS NR) messages. On obtaining this message, the RPL owner blocks the RPL port and sends RAPS no request, root blocked (RAPS NR, RB) messages. This causes all other nodes, other than the RPL owner in the ring, to unblock all blocked ports. The ERP protocol is robust enough to work for both unidirectional failure and multiple link failure scenarios in a ring topology.

A G.8032 ring supports these basic operator administrative commands:

- **Force switch (FS)**—Allows operator to forcefully block a particular ring-port.
  - Effective even if there is an existing SF condition
  - Multiple FS commands for ring supported
  - May be used to allow immediate maintenance operations

- **Manual switch (MS)**—Allows operator to manually block a particular ring-port.
  - Ineffective in an existing FS or SF condition
  - Overridden by new FS or SF conditions
  - Multiple MS commands cancel all MS commands

- **Clear**— Cancels an existing FS or MS command on the ring-port
  - Used (at RPL Owner) to clear non-revertive mode

A G.8032 ring can support multiple instances. An instance is a logical ring running over a physical ring. Such instances are used for various reasons, such as load balancing VLANs over a ring. For example, odd VLANs may go in one direction of the ring, and even VLANs may go in the other direction. Specific VLANs can be configured under only one instance. They cannot overlap multiple instances. Otherwise, data traffic or RAPS packet can cross logical rings, and that is not desirable.

G.8032 ERP provides a new technology that relies on line status and Connectivity Fault Management (CFM) to detect link failure. By running CFM Continuity Check Messages (CCM) messages at an interval of 100ms, it is possible to achieve SONET-like switching time performance and loop free traffic.

For more information about Ethernet Connectivity Fault Management (CFM) and Ethernet Fault Detection (EFD) configuration, refer to the Configuring Ethernet OAM on the Cisco ASR 9000 Series Router module in the Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide.

**Timers**

G.8032 ERP specifies the use of different timers to avoid race conditions and unnecessary switching operations:

- **Delay Timers**—used by the RPL Owner to verify that the network has stabilized before blocking the RPL
  - After SF condition, Wait-to-Restore (WTR) timer is used to verify that SF is not intermittent. The WTR timer can be configured by the operator, and the default time interval is 5 minutes. The time interval ranges from 1 to 12 minutes.
  - After FS/MS command, Wait-to-Block timer is used to verify that no background condition exists.

**Note**

Wait-to-Block timer may be shorter than the Wait-to-Restore timer.
Information About Implementing Multipoint Layer 2 Services

- **Guard Timer**—used by all nodes when changing state; it blocks latent outdated messages from causing unnecessary state changes. The Guard timer can be configured and the default time interval is 500 ms. The time interval ranges from 10 to 2000 ms.

- **Hold-off timers**—used by underlying Ethernet layer to filter out intermittent link faults. The hold-off timer can be configured and the default time interval is 0 seconds. The time interval ranges from 0 to 10 seconds.
  - Faults are reported to the ring protection mechanism, only if this timer expires.

### Single Link Failure

Figure 8 represents protection switching in case of a single link failure.

**Figure 8**  
**G.8032 Single Link Failure**

Figure 8 represents an Ethernet ring composed of seven Ethernet ring nodes. The RPL is the ring link between Ethernet ring nodes A and G. In these scenarios, both ends of the RPL are blocked. Ethernet ring node G is the RPL owner node, and Ethernet ring node A is the RPL neighbor node.

These symbols are used:

- **Message source**
- **R-APS channel blocking**
- **Client channel blocking**
- **Node ID**

This sequence describes the steps in the single link failure, represented in Figure 8:

1. Link operates in the normal condition.
2. A failure occurs.
3. Ethernet ring nodes C and D detect a local Signal Failure condition and after the holdoff time interval, block the failed ring port and perform the FDB flush.
4. Ethernet ring nodes C and D start sending RAPS (SF) messages periodically along with the (Node ID, BPR) pair on both ring ports, while the SF condition persists.

5. All Ethernet ring nodes receiving an RAPS (SF) message perform FDB flush. When the RPL owner node G and RPL neighbor node A receive an RAPS (SF) message, the Ethernet ring node unblocks its end of the RPL and performs the FDB flush.

6. All Ethernet ring nodes receiving a second RAPS (SF) message perform the FDB flush again; this is because of the Node ID and BPR-based mechanism.

7. Stable SF condition—RAPS (SF) messages on the Ethernet Ring. Further RAPS (SF) messages trigger no further action.

Figure 9 represents reversion in case of a single link failure.

**Figure 9**  
*Single link failure Recovery (Revertive operation)*

This sequence describes the steps in the single link failure recovery, as represented in Figure 9:

1. Link operates in the stable SF condition.

2. Recovery of link failure occurs.

3. Ethernet ring nodes C and D detect clearing of signal failure (SF) condition, start the guard timer and initiate periodical transmission of RAPS (NR) messages on both ring ports. (The guard timer prevents the reception of RAPS messages).

4. When the Ethernet ring nodes receive an RAPS (NR) message, the Node ID and BPR pair of a receiving ring port is deleted and the RPL owner node starts the WTR timer.

5. When the guard timer expires on Ethernet ring nodes C and D, they may accept the new RAPS messages that they receive. Ethernet ring node D receives an RAPS (NR) message with higher Node ID from Ethernet ring node C, and unblocks its non-failed ring port.

6. When WTR timer expires, the RPL owner node blocks its end of the RPL, sends RAPS (NR, RB) message with the (Node ID, BPR) pair, and performs the FDB flush.
7. When Ethernet ring node C receives an RAPS (NR, RB) message, it removes the block on its blocked ring ports, and stops sending RAPS (NR) messages. On the other hand, when the RPL neighbor node A receives an RAPS (NR, RB) message, it blocks its end of the RPL. In addition to this, Ethernet ring nodes A to F perform the FDB flush when receiving an RAPS (NR, RB) message, due to the existence of the Node ID and BPR based mechanism.

**Flow Aware Transport Pseudowire (FAT PW)**

Routers typically loadbalance traffic based on the lower most label in the label stack which is the same label for all flows on a given pseudowire. This can lead to asymmetric loadbalancing. The flow, in this context, refers to a sequence of packets that have the same source and destination pair. The packets are transported from a source provider edge (PE) to a destination PE.

Flow-Aware Transport Pseudowires (FAT PW) provide the capability to identify individual flows within a pseudowire and provide routers the ability to use these flows to loadbalance traffic. FAT PWs are used to loadbalance traffic in the core when equal cost multipaths (ECMP) are used. A flow label is created based on indivisible packet flows entering a pseudowire; and is inserted as the lower most label in the packet. Routers can use the flow label for loadbalancing which provides a better traffic distribution across ECMP paths or link-bundled paths in the core.

Figure 10 shows a FAT PW with two flows distributing over ECMPs and bundle links.

**Figure 10  FAT PW with two flows distributing over ECMPs and Bundle-Links**

An additional label is added to the stack, called the flow label, which contains the flow information of a virtual circuit (VC). A flow label is a unique identifier that distinguishes a flow within the PW, and is derived from source and destination MAC addresses, and source and destination IP addresses. The flow label contains the end of label stack (EOS) bit set and inserted after the VC label and before the control word (if any). The ingress PE calculates and forwards the flow label to load balance on ECMPs or bundles. The egress PE removes the flow label and can use it for bundle AC load-balance.

All core routers perform load balancing based on the flow-label in the FAT PW. Therefore, it is possible to distribute flows over ECMPs and link bundles.
**Pseudowire Headend**

Pseudowires (PWs) enable payloads to be transparently carried across IP/MPLS packet-switched networks (PSNs). PWs are regarded as simple and manageable lightweight tunnels for returning customer traffic into core networks. Service providers are now extending PW connectivity into the access and aggregation regions of their networks.

Pseudowire Headend (PWHE) is a technology that allows termination of access pseudowires (PWs) into a Layer 3 (VRF or global) domain or into a Layer 2 domain. PWs provide an easy and scalable mechanism for tunneling customer traffic into a common IP/MPLS network infrastructure. PWHE allows customers to provision features such as QoS access lists (ACL), L3VPN on a per PWHE interface basis, on a service Provider Edge (PE) router.

**Benefits of PWHE**

Some of the benefits of implementing PWHE are:

- dissociates the customer facing interface (CFI) of the service PE from the underlying physical transport media of the access or aggregation network
- reduces capex in the access or aggregation network and service PE
- distributes and scales the customer facing Layer 2 UNI interface set
- implements a uniform method of OAM functionality
- providers can extend or expand the Layer 3 service footprints
- provides a method of terminating customer traffic into a next generation network (NGN)

**Restrictions**

Pseudowire Headend is supported only on the following hardware:

- ASR 9000 Enhanced Ethernet Line Cards
- SE (Service Edge) variant of ASR 9000 Ethernet Line Cards

**L2VPN over GRE**

To transport an IP packet over a generic routing encapsulation (GRE) tunnel, the system first encapsulates the original IP packet with a GRE header. The encapsulated GRE packet is encapsulated once again by an outer transport header that is used to forward the packet to its destination. Figure 11 captures how GRE encapsulation over an IP transport network takes place.

*Figure 11  GRE Encapsulation*

<table>
<thead>
<tr>
<th>Original IP Packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original IP Hdr</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>New IP Packet after Encapsulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>New IP Hdr</td>
</tr>
</tbody>
</table>
Note

In the new IP packet, new payload is similar to the original IP packet. Additionally, the new IP header (New IP Hdr) is similar to the tunnel IP header which in turn is similar to the transport header.

When a GRE tunnel endpoint decapsulates a GRE packet, it further forwards the packet based on the payload type. For example, if the payload is a labeled packet then the packet is forwarded based on the virtual circuit (VC) label or the VPN label for L2VPN and L3VPN respectively.

L2VPN over GRE Restrictions

Some of the restrictions that you must consider while configuring L2VPN over GRE:

- For VPLS flow-based load balancing scenario, the GRE tunnel is pinned down to outgoing path based on tunnel source or destination cyclic redundancy check (CRC). Unicast and flood traffic always takes the same physical path for a given GRE tunnel.
- Ingress attachment circuit must be an ASR 9000 Enhanced Ethernet Line Card for L2VPN over GRE. Additionally, GRE tunnel destination should be reachable only on an ASR 9000 Enhanced Ethernet Line Card.
- The L2VPN over GRE feature is not supported on the ASR 9000 Ethernet Line Card or Cisco ASR 9000 Series SPA Interface Processor-700 line cards as the ingress attachment circuit and GRE destination is reachable over GRE.
- Pseudowire over TE over GRE scenario is not supported.
- Preferred Path Limitations:
  - When you configure GRE as a preferred path, egress features are not supported under the GRE tunnel (Egress ACL).
  - VCCV ping or traceroute are not supported for preferred path.
  - Preferred path is supported only for pseudowires configured in a provider egde (PE) to PE topology.

GRE Deployment Scenarios

In an L2VPN network, you can deploy GRE in the following scenarios:

- Configuring GRE tunnel between provider edge (PE) to PE routers
- Configuring GRE tunnel between P to P routers
- Configuring GRE tunnel between P to PE routers

The following diagrams depict the various scenarios:
Information About Implementing Multipoint Layer 2 Services

Chapter Implementing Multipoint Layer 2 Services

Figure 12 GRE tunnel configured between PE to PE routers

![Diagram of GRE tunnel configured between PE to PE routers]

Figure 13 GRE tunnel configured between P to P routers

![Diagram of GRE tunnel configured between P to P routers]

Figure 14 GRE tunnel configured between P to PE routers

![Diagram of GRE tunnel configured between P to PE routers]
Note

These deployment scenarios are applicable to VPWS and VPLS.

**GRE Tunnel as Preferred Path**

Preferred tunnel path feature allows you to map pseudowires to specific GRE tunnels. Attachment circuits are cross-connected to GRE tunnel interfaces instead of remote PE router IP addresses (reachable using IGP or LDP). Using preferred tunnel path, it is always assumed that the GRE tunnel that transports the L2 traffic runs between the two PE routers (that is, its head starts at the imposition PE router and terminates on the disposition PE router).
How to Implement Multipoint Layer 2 Services

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

- Configuring a Bridge Domain, page 258
- Configuring Layer 2 Security, page 274
- Configuring a Layer 2 Virtual Forwarding Instance, page 278
- Configuring the MAC Address-related Parameters, page 290
- Configuring an Attachment Circuit to the AC Split Horizon Group, page 305
- Adding an Access Pseudowire to the AC Split Horizon Group, page 307
- Configuring VPLS with BGP Autodiscovery and Signaling, page 308
- Configuring VPLS with BGP Autodiscovery and LDP Signaling, page 311
- Configuring G.8032 Ethernet Ring Protection, page 314
- Configuring Flow Aware Transport Pseudowire, page 323
- Configuring Pseudowire Headend, page 329
- Configuring L2VPN over GRE, page 341

Configuring a Bridge Domain

These topics describe how to configure a bridge domain:

- Creating a Bridge Domain, page 258
- Configuring a Pseudowire, page 260
- Associating Members with a Bridge Domain, page 263
- Configuring Bridge Domain Parameters, page 265
- Disabling a Bridge Domain, page 268
- Blocking Unknown Unicast Flooding, page 270
- Changing the Flood Optimization Mode, page 271

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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# l2vpn RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> bridge group bridge-group-name</td>
<td>Creates a bridge group that can contain bridge domains, and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group csco RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> bridge-domain bridge-domain-name</td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> end or commit</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# end or RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit</td>
<td></td>
</tr>
</tbody>
</table>

- **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. `dhcp ipv4 snoop profile [dhcp_snoop_profile_name]`
9. `end`
   or
   `commit`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1**
  `configure` | Enters global configuration mode. |
  **Example:**
  `RP/0/RSP0/CPU0:router# configure` |
| **Step 2**
  `l2vpn` | Enters L2VPN configuration mode. |
  **Example:**
  `RP/0/RSP0/CPU0:router(config)# l2vpn
  RP/0/RSP0/CPU0:router(config-l2vpn)#` |
| **Step 3**
  `bridge group bridge-group-name` | Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain. |
  **Example:**
  `RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco
  RP/0/RSP0/CPU0:router(config-l2vpn-bg)#` |
| **Step 4**
  `bridge-domain bridge-domain-name` | Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode. |
  **Example:**
  `RP/0/RSP0/CPU0:router(config-l2vpn-bg)#
  bridge-domain abc
  RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#` |
### Command or Action

#### Step 5

```bash
vfi {vfi-name}
```

**Example:**

```bash
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#
```

**Purpose:**

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

```bash
exit
```

**Example:**

```bash
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
```

**Purpose:**

Exits the current configuration mode.

#### Step 7

```bash
neighbor {A.B.C.D} {pw-id value}
```

**Example:**

```bash
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# neighbor 10.1.1.2 pw-id 1000
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)#
```

**Purpose:**

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.

**Note**

- `A.B.C.D` can be a recursive or non-recursive prefix.
- Use the `pw-id` keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.
### How to Implement Multipoint Layer 2 Services

**Step 8**

**Command or Action**: `dhcp ipv4 snoop profile (dhcp_snoop_profile_name)`

**Example**:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)# dhcp ipv4 snoop profile profile1

**Purpose**: Enables DHCP snooping on the bridge, and attaches a DHCP snooping profile.

**Step 9**

**Command or Action**: `end` or `commit`

**Example**:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)# end  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)# commit

**Purpose**: 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. These 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 type interface-path-id`
6. `static-mac-address {MAC-address}`
7. `end`
   or
   `commit`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td><code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>Step 2</td>
<td><code>l2vpn</code></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td>Step 3</td>
<td><code>bridge group bridge-group-name</code></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td>Step 4</td>
<td><code>bridge-domain bridge-domain-name</code></td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action | Purpose
--- | ---
**Step 5**
`interface type interface-path-id` | Enters interface configuration mode and 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.

**Example:**
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface
GigabitEthernet 0/4/0/0
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#

**Step 6**
`static-mac-address MAC-address` | Configures the static MAC address to associate a remote MAC address with a pseudowire or any other bridge interface.

**Example:**
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#
static-mac-address 1.1.1

**Step 7**
end
or
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 bridge domain parameters, associate these 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.
- **Dynamic ARP Inspection (DAI)**—Ensures only valid ARP requests and responses are relayed.
- **IP SourceGuard (IPSG)**—Enables source IP address filtering on a Layer 2 port.

### Note

To verify if the DAI and IPSG features are working correctly, look up the packets dropped statistics for DAI and IPSG violation. The packet drops statistics can be viewed in the output of the `show l2vpn bridge-domain bd-name <> detail` command.

## 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. dynamic-arp-inspection [address-validation | disable | logging]
8. ip-source-guard logging
9. end
   or
   commit

## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>configure</td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

<table>
<thead>
<tr>
<th><strong>Step 2</strong></th>
<th>Enters L2VPN configuration mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td>12vpn</td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router(config)# 12vpn
RP/0/RSP0/CPU0:router(config-12vpn)#
```
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 3</strong>  bridge group bridge-group-name</td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong>  bridge-domain bridge-domain-name</td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong>  flooding disable</td>
<td>Configures flooding for traffic at the bridge domain level or at the bridge port level.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# flooding disable</td>
<td></td>
</tr>
<tr>
<td><strong>Step 6</strong>  mtu bytes</td>
<td>Adjusts the maximum packet size or maximum transmission unit (MTU) size for the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# mtu 1000</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong>  dynamic-arp-inspection  (address-validation</td>
<td>disable</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# dynamic-arp-inspection</td>
<td></td>
</tr>
</tbody>
</table>

**Note**  You can configure dynamic ARP inspection under the bridge domain or the bridge port.
### Command or Action

**Step 8**

```
ip-source-guard logging
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
ip-source-guard logging
```

**Purpose:**

Enters the IP source guard configuration submode and enables source IP address filtering on a Layer 2 port.

You can enable IP source guard under the bridge domain or the bridge port. By default, bridge ports under a bridge inherit the IP source guard configuration from the parent bridge.

By default, IP source guard is disabled on the bridges.

**Step 9**

```
end
```

or

```
commit
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# end
```

or

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit
```

**Purpose:**

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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>configure</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>l2vpn</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td>bridge group bridge-group-name</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td>bridge-domain bridge-domain-name</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
</tbody>
</table>
#### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command/Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>shutdown</td>
<td>Shuts down a bridge domain to bring the bridge and all attachment circuits and pseudowires under it to admin down state.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
```

<table>
<thead>
<tr>
<th>Step</th>
<th>Command/Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| 6    | end or 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. |

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# end

RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit
```
## Blocking Unknown Unicast Flooding

Perform this task to disable flooding of unknown unicast traffic at the bridge domain level.

You can disable flooding of unknown unicast traffic at the bridge domain, bridge port or access pseudowire levels. By default, unknown unicast traffic is flooded to all ports in the bridge domain.

### Note

If you disable flooding of unknown unicast traffic on the bridge domain, all ports within the bridge domain inherit this configuration. You can configure the bridge ports to override the bridge domain configuration.

### SUMMARY STEPS

1. configure
2. l2vpn
3. bridge group bridge-group-name
4. bridge-domain bridge-domain-name
5. flooding unknown-unicast disable
6. end
   or
   commit

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router# configure</td>
</tr>
<tr>
<td>Enters global configuration mode.</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>l2vpn</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td>Enters L2VPN configuration mode.</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>bridge group bridge-group-name</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>bridge-domain bridge-domain-name</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td>Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.</td>
<td></td>
</tr>
</tbody>
</table>
### How to Implement Multipoint Layer 2 Services

**SUMMARY STEPS**

1. configure
2. l2vpn
3. bridge group <br>`bridge-group name`<br>
4. bridge-domain <br>`bridge-domain name`
5. flood mode convergence-optimized
6. end <br>or <br>commit

#### Changing the Flood Optimization Mode

Perform this task to change the flood optimization mode under the bridge domain:

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>flooding unknown-unicast disable</td>
<td>Enables flooding of unknown unicast traffic at the bridge domain level.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# flooding unknown-unicast disable</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>end or commit</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# end or RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit</td>
<td></td>
</tr>
</tbody>
</table>

When you issue the **end** command, the system prompts you to commit changes:

- 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.
# How to Implement Multipoint Layer 2 Services

## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td><strong>configure</strong></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>configure</code></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router#</td>
<td></td>
</tr>
<tr>
<td>configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td><strong>l2vpn</strong></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>l2vpn</code></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router#</td>
<td></td>
</tr>
<tr>
<td>(config)#</td>
<td></td>
</tr>
<tr>
<td><code>l2vpn</code></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router#</td>
<td></td>
</tr>
<tr>
<td>(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td></td>
</tr>
<tr>
<td><strong>bridge group</strong></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td>bridge-group-name</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>bridge group cisco</code></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router#</td>
<td></td>
</tr>
<tr>
<td>(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td></td>
</tr>
<tr>
<td><strong>bridge-domain</strong></td>
<td>Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td>bridge-domain-name</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>bridge-domain abc</code></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router#</td>
<td></td>
</tr>
<tr>
<td>(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

| Step 5 | flood mode convergence-optimized |

**Example:**

RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
flood mode convergence-optimized

Changes the default flood optimization mode from Bandwidth Optimization Mode to Convergence Mode.

| Step 6 | end or commit |

**Example:**

RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# end

or

RP/0/RSP0/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 Layer 2 Security

These topics describe how to configure Layer 2 security:

- Enabling Layer 2 Security, page 274
- Attaching a Dynamic Host Configuration Protocol Profile, page 275

Enabling Layer 2 Security

Perform this task to enable Layer 2 port security on a bridge.

SUMMARY STEPS

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

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>
| **Example:**
  RP/0/RSP0/CPU0:router# configure | |
| **Step 2** l2vpn | Enters L2VPN configuration mode. |
| **Example:**
  RP/0/RSP0/CPU0:router(config)# l2vpn
  RP/0/RSP0/CPU0:router(config-l2vpn)# | |
| **Step 3** bridge group bridge-group-name | Assigns each network interface to a bridge group and enters L2VPN bridge group configuration mode. |
| **Example:**
  RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco
  RP/0/RSP0/CPU0:router(config-l2vpn-bg)# | |
| **Step 4** bridge-domain bridge-domain-name | Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode. |
| **Example:**
  RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc
  RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# | |
## How to Implement Multipoint Layer 2 Services

### Summary Steps

#### Step 5

**Command or Action**

`security`

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
security
```

**Purpose**

Enables Layer 2 port security on a bridge.

#### Step 6

**Command or Action**

`end` or `commit`

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
end
```

**Purpose**

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 a Dynamic Host Configuration Protocol Profile

Perform this task to enable DHCP snooping on a bridge and to attach a DHCP snooping profile to a bridge.

### Summary Steps

1. `configure`
2. `l2vpn`
3. `bridge group bridge-group-name`
4. `bridge-domain bridge-domain-name`
5. `dhcp ipv4 snoop {profile profile-name}`
6. `end` or `commit`
## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# l2vpn RP/0/RSP0/CPU0:router(config-12vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> bridge group <strong>bridge-group-name</strong></td>
<td>Assigns each network interface to a bridge group and enters L2VPN bridge group configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-12vpn)# bridge group cisco RP/0/RSP0/CPU0:router(config-12vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> bridge-domain <strong>bridge-domain-name</strong></td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-12vpn-bg)# bridge-domain abc RP/0/RSP0/CPU0:router(config-12vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Step 5: <code>dhcp ipv4 snoop (profile profile-name)</code></td>
<td>Enables DHCP snooping on a bridge and attaches DHCP snooping profile to the bridge. Use the profile keyword to attach a DHCP profile. The profile-name argument is the profile name for DHCPv4 snooping.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# dhcp ipv4 snoop profile attach</td>
</tr>
</tbody>
</table>

| Step 6: `end` or `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]: |
| Example:                              | RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# end or commit               |
| RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit | - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |
| RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit | - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |
| RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit | - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |
| RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# commit | - 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 278
- Associating Pseudowires with the Virtual Forwarding Instance, page 280
- Associating a Virtual Forwarding Instance to a Bridge Domain, page 282
- Attaching Pseudowire Classes to Pseudowires, page 284
- Configuring Any Transport over Multiprotocol Pseudowires By Using Static Labels, page 286
- Disabling a Virtual Forwarding Instance, page 288

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**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td><code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><code>RP/0/RSP0/CPU0:router# configure</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td><code>l2vpn</code></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><code>RP/0/RSP0/CPU0:router(config)# l2vpn</code></td>
<td></td>
</tr>
<tr>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn)#</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td></td>
</tr>
<tr>
<td><code>bridge group bridge-group-name</code></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</code></td>
<td></td>
</tr>
<tr>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</code></td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td>bridge-domain bridge-domain-name</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td>Configures virtual forwarding interface (VFI) parameters and enters L2VPN bridge group bridge domain VFI configuration mode.</td>
</tr>
<tr>
<td>vfi (vfi-name)</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1 RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#</td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td>end or</td>
<td>When you issue the <strong>end</strong> command, the system prompts you to commit changes:</td>
</tr>
<tr>
<td>commit</td>
<td>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# end or RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# commit</td>
</tr>
<tr>
<td></td>
<td>- Entering <strong>yes</strong> saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>- Entering <strong>no</strong> exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>- Entering <strong>cancel</strong> leaves the router in the current configuration session without exiting or committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>- Use the <strong>commit</strong> command to save the configuration changes to the running configuration file and remain within the configuration session.</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# l2vpn RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> bridge group bridge-group-name</td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> bridge-domain bridge-domain-name</td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> vfi {vfi-name}</td>
<td>Configures virtual forwarding interface (VFI) parameters and enters L2VPN bridge group bridge domain VFI configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1 RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

**Step 6**

```
neighbor (A.B.C.D) (pw-id value)
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#
neighbor 10.1.1.2 pw-id 1000
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#
```

**Purpose**

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/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#
end
or
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#
commit
```

**Purpose**

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.
How to Implement Multipoint Layer 2 Services

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**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router# configure</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>l2vpn</code></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config)# l2vpn</code> <code>RP/0/RSP0/CPU0:router(config-l2vpn)#</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>bridge group bridge-group-name</code></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</code> <code>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>bridge-domain bridge-domain-name</code></td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc</code> <code>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> <code>vfi {vfi-name}</code></td>
<td>Configures virtual forwarding interface (VFI) parameters and enters L2VPN bridge group bridge domain VFI configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1</code> <code>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#</code></td>
<td></td>
</tr>
</tbody>
</table>
### How to Implement Multipoint Layer 2 Services

#### Step 6

**neighbor** *(A.B.C.D) {pw-id value}*

**Example:**
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#
neighbor 10.1.1.2 pw-id 1000
```

- 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

**static-mac-address** *(MAC-address)*

**Example:**
```
RP/0/RSP0/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**

**commit**

**Example:**
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#
end
```

- 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.
How to Implement Multipoint Layer 2 Services

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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>l2vpn</code></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>bridge group bridge-group-name</code></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>bridge-domain bridge-domain-name</code></td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> <code>vfi {vfi-name}</code></td>
<td>Configures virtual forwarding interface (VFI) parameters and enters L2VPN bridge group bridge domain VFI configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#</td>
<td></td>
</tr>
</tbody>
</table>
## How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 6** neighbor \(A.B.C.D\) \{pw-id value\} | 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. |

**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi)#  
neighbor 10.1.1.2 pw-id 1000  
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)#

| **Step 7** pw-class \{class-name\} | Configures the pseudowire class template name to use for the pseudowire. |

**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)#  
pw-class canada

| **Step 8** end or 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. |

**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)#  
end  
or  
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)#  
commit
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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>

Example:
RP/0/RSP0/CPU0:router# configure

**Step 2** l2vpn
Enters L2VPN configuration mode.

Example:
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)#

**Step 3** bridge group bridge-group-name
Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

Example:
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco
RP/0/RSP0/CPU0:router(config-l2vpn-bg)#

**Step 4** bridge-domain bridge-domain-name
Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg)#
bridge-domain abc
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
### Command or Action

**Step 5**  
`vfi (vfi-name)`

**Example:**  
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#
```

**Purpose:**  
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/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# neighbor 10.1.1.2 pw-id 1000  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)#
```

**Purpose:**  
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**  
`mpls static label {local value} {remote value}`

**Example:**  
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# mpls static label local 800 remote 500
```

**Purpose:**  
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`  

**Example:**  
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# end
```

**Purpose:**  
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**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>configure</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Enters global configuration mode.</strong></td>
</tr>
<tr>
<td>RP/0/RSP0/CP00:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><strong>l2vpn</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Enters L2VPN configuration mode.</strong></td>
</tr>
<tr>
<td>RP/0/RSP0/CP00:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CP00:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><strong>bridge group bridge-group-name</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</strong></td>
</tr>
<tr>
<td>RP/0/RSP0/CP00:router(config-l2vpn)# bridge group cisco</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CP00:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td><strong>bridge-domain bridge-domain-name</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</strong></td>
</tr>
<tr>
<td>RP/0/RSP0/CP00:router(config-l2vpn-bg)# bridge-domain abc</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CP00:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>vfi (vfi)-name)</td>
<td>Configures virtual forwarding interface (VFI) parameters and enters L2VPN bridge group bridge domain VFI configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)#</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>shutdown</td>
<td>Disables the virtual forwarding interface (VFI).</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# shutdown</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>end</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td></td>
<td>or commit</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# end</td>
<td></td>
</tr>
<tr>
<td></td>
<td>or commit</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# commit</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>show l2vpn bridge-domain [detail]</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# show l2vpn bridge-domain detail</td>
<td></td>
</tr>
</tbody>
</table>
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 290
- Enabling the MAC Address Withdrawal, page 293
- Configuring the MAC Address Limit, page 295
- Configuring the MAC Address Aging, page 298
- Disabling MAC Flush at the Bridge Port Level, page 301
- Configuring MAC Address Security, page 303

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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# configure</td>
</tr>
<tr>
<td>Step 2 l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
</tr>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td><code>bridge group bridge-group-name</code></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco&lt;br&gt;RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>bridge-domain bridge-domain-name</code></td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-l2vpn-bg)#&lt;br&gt;bridge-domain abc&lt;br&gt;RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td><code>mac</code></td>
<td>Enters L2VPN bridge group bridge domain MAC configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# mac&lt;br&gt;RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)#</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td><code>learning disable</code></td>
<td>Disables MAC learning at the bridge domain level.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# learning disable</td>
<td></td>
</tr>
</tbody>
</table>
### Step 7

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>end</code> or <code>commit</code></td>
<td>Saves configuration changes.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# end
```

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# commit
```

- 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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>show l2vpn bridge-domain [detail]</code></td>
<td>Displays the details that the MAC address source-based learning is disabled on the bridge.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router# show l2vpn bridge-domain detail
```
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**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# l2vpn RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> bridge group <em>bridge-group-name</em></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> bridge-domain <em>bridge-domain-name</em></td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> mac</td>
<td>Enters L2VPN bridge group bridge domain MAC configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# mac RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)#</td>
<td></td>
</tr>
</tbody>
</table>
How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 6</strong></td>
<td></td>
</tr>
<tr>
<td>withdrawal</td>
<td>Enables the MAC address withdrawal for a specified bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# withdrawal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td></td>
</tr>
<tr>
<td>end</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td>or</td>
<td></td>
</tr>
<tr>
<td>commit</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# end</td>
<td></td>
</tr>
<tr>
<td>or</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# commit</td>
<td></td>
</tr>
<tr>
<td><strong>Step 8</strong></td>
<td></td>
</tr>
<tr>
<td>show l2vpn bridge-domain [detail]</td>
<td>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.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>P/0/RSP0/CPU0:router# show l2vpn bridge-domain detail</td>
<td></td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>
| configure        | Example:  
|                  | RP/0/RSP0/CPU0:router# configure |
| **Step 2**       | Enters L2VPN configuration mode. |
| l2vpn            | Example:  
|                  | RP/0/RSP0/CPU0:router(config)# l2vpn |
|                  | RP/0/RSP0/CPU0:router(config-l2vpn)# |
| **Step 3**       | Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain. |
| bridge group     | Example:  
| bridge-group-name| RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco |
|                  | RP/0/RSP0/CPU0:router(config-l2vpn-bg)# |
| **Step 4**       | Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode. |
| bridge-domain    | Example:  
| bridge-domain-name| RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc |
|                  | RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# |
# Implementing Multipoint Layer 2 Services

## How to Implement Multipoint Layer 2 Services

### Step 5: `mac`

**Example:**
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# mac
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)#
```

Enters L2VPN bridge group bridge domain MAC configuration mode.

### Step 6: `limit`

**Example:**
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# limit
RP/0/RSP0/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/RSP0/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/RSP0/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.

### Step 9: `notification {both | none | trap}`

**Example:**
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac-limit)#
nomination both
```

Specifies the type of notification that is sent when the number of learned MAC addresses exceeds the configured limit.
### Command or Action

<table>
<thead>
<tr>
<th>Step 10</th>
<th>Command or Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>end or commit</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac-limit)#
end
```

```
RP/0/RSP0/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.

<table>
<thead>
<tr>
<th>Step 11</th>
<th>Command or Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>show l2vpn bridge-domain [detail]</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router# show l2vpn bridge-domain
detail
```
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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>Step 2 l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td></td>
</tr>
<tr>
<td>Step 3 bridge group bridge-group-name</td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group cisco</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td>Step 4 bridge-domain bridge-domain-name</td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain abc</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

#### Step 5

mac

Example:
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# mac
```

Enters L2VPN bridge group bridge domain MAC configuration mode.

#### Step 6

aging

Example:
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# aging
```

Enters the MAC aging configuration submode to set the aging parameters such as time and type.

The maximum MAC age for ASR 9000 Ethernet and ASR 9000 Enhanced Ethernet line cards is two hours.

#### Step 7

time \{seconds\}

Example:
```
RP/0/RSP0/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.

#### Step 8

type \{absolute | inactivity\}

Example:
```
RP/0/RSP0/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.
How to Implement Multipoint Layer 2 Services

### Step 9

**Command or Action**

- `end`
- `commit`

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac-aging)# end
```

- `commit`

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac-aging)# commit
```

**Purpose**

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

**Command or Action**

- `show l2vpn bridge-domain [detail]`

**Example:**

```
RP/0/RSP0/CPU0:router# show l2vpn bridge-domain
detail
```

**Purpose**

Displays the details about the aging fields.
### Disabling MAC Flush at the Bridge Port Level

Perform this task to disable the MAC flush at the bridge domain level.

You can disable the MAC flush at the bridge domain, bridge port or access pseudowire levels. By default, the MACs learned on a specific port are immediately flushed, when that port becomes nonfunctional.

#### SUMMARY STEPS

1. `configure`
2. `l2vpn`
3. `bridge group bridge-group name`
4. `bridge-domain bridge-domain name`
5. `mac`
6. `port-down flush disable`
7. `end`
   
   or
   
   `commit`

#### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router# configure</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>l2vpn</code></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config)# l2vpn</code> <code>RP/0/RSP0/CPU0:router(config-l2vpn)#$</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>bridge group bridge-group-name</code></td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config-l2vpn)#$ bridge group cisco</code> <code>RP/0/RSP0/CPU0:router(config-12vpn-bg)#$</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>bridge-domain bridge-domain-name</code></td>
<td>Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>RP/0/RSP0/CPU0:router(config-12vpn-bg)#$ bridge-domain abc</code> <code>RP/0/RSP0/CPU0:router(config-12vpn-bg-bd)#$</code></td>
<td></td>
</tr>
</tbody>
</table>
## How to Implement Multipoint Layer 2 Services

### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>5</strong></td>
<td><code>mac</code></td>
<td>Enters l2vpn bridge group bridge domain MAC configuration mode.</td>
</tr>
</tbody>
</table>

**Example:**

RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# `mac`  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)#

| **6** | `port-down flush disable` | Disables MAC flush when the bridge port becomes nonfunctional. |

**Example:**

RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-mac)# `port-down flush disable`

| **7** | `end` or `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 MAC Address Security

Perform this task to configure MAC address security.

## SUMMARY STEPS

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

## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-12vpn)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> bridge group bridge-group-name</td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-12vpn)# bridge group cisco</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-12vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> bridge-domain bridge-domain-name</td>
<td>Establishes a bridge domain and enters l2vpn bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-12vpn-bg)# bridge-domain abc</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-12vpn-bg-bd)#</td>
<td></td>
</tr>
</tbody>
</table>
## How to Implement Multipoint Layer 2 Services

### Step 5

**Command or Action**: `neighbor (A.B.C.D) {pw-id value}

**Example**:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# neighbor 10.1.1.2 pw-id 1000
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)#

**Purpose**: 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 6

**Command or Action**: `mac`

**Example**:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)# mac
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw-mac)#

**Purpose**: Enters l2vpn bridge group bridge domain MAC configuration mode.

### Step 7

**Command or Action**: `secure [action | disable | logging]`

**Example**:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw)# secure
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw-secure)#

**Purpose**: Enters MAC secure configuration mode.  
By default, bridge ports (interfaces and access pseudowires) under a bridge inherit the security configuration from the parent bridge.  
**Note**: Once a bridge port goes down, a `clear` command must be issued to bring the bridge port up.

### Step 8

**Command or Action**: `end` or `commit`

**Example**:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw-secure)# end
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-pw-secure)# commit

**Purpose**: Saves configuration changes.  
- When you issue the `end` command, the system prompts you to commit changes:  
  ```plaintext
  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 an Attachment Circuit to the AC Split Horizon Group

These steps show how to add an interface to the split horizon group for attachment circuits (ACs) under a bridge domain.

**SUMMARY STEPS**

1. configure
2. l2vpn
3. bridge group *bridge-group-name*
4. bridge-domain *bridge-domain-name*
5. interface *type instance*
6. split-horizon group
7. commit
8. end
9. show l2vpn bridge-domain detail

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>bridge group <em>bridge-group-name</em></td>
<td>Enters configuration mode for the named bridge group.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group metroA</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>bridge-domain <em>bridge-domain-name</em></td>
<td>Enters configuration mode for the named bridge domain.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain east</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>interface <em>type instance</em></td>
<td>Enters configuration mode for the named interface.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet0/1/0/6</td>
<td></td>
</tr>
</tbody>
</table>
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 6 <em>split-horizon group</em></td>
<td>Adds this interface to the split horizon group for ACs. Only one split horizon group for ACs for a bridge domain is supported.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-ac)#</td>
<td>split-horizon group</td>
</tr>
<tr>
<td>Step 7 <em>commit</em></td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-ac)#</td>
<td>commit</td>
</tr>
<tr>
<td>Step 8 <em>end</em></td>
<td>Returns to EXEC mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-ac)#</td>
<td>end</td>
</tr>
<tr>
<td>Step 9 <em>show l2vpn bridge-domain detail</em></td>
<td>Displays information about bridges, including whether each AC is in the AC split horizon group or not.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# show l2vpn bridge-domain detail</td>
<td></td>
</tr>
</tbody>
</table>
Adding an Access Pseudowire to the AC Split Horizon Group

These steps show how to add an access pseudowire as a member to the split horizon group for attachment circuits (ACs) under a bridge domain.

**SUMMARY STEPS**

1. `configure`
2. `l2vpn`
3. `bridge group bridge-group-name`
4. `bridge-domain bridge-domain-name`
5. `neighbor A.B.C.D pw-id pseudowire-id`
6. `split-horizon group`
7. `commit`
8. `end`
9. `show l2vpn bridge-domain detail`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><code>configure</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router# configure</code></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><code>l2vpn</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config)# l2vpn</code></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><code>bridge group bridge-group-name</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group metroA</code></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td><code>bridge-domain bridge-domain-name</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config-12vpn-bg)# bridge-domain east</code></td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td><code>neighbor A.B.C.D pw-id pseudowire-id</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/RSP0/CPU0:router(config-12vpn-bg-bd)# neighbor 10.2.2.2 pw-id 2000</code></td>
</tr>
</tbody>
</table>
How to Implement Multipoint Layer 2 Services

Configuring VPLS with BGP Autodiscovery and Signaling

Perform this task to configure BGP-based autodiscovery and signaling.

To locate documentation for the commands used in this configuration, refer to the Multipoint Layer 2 Services Commands module in the Cisco ASR 9000 Series Aggregation Services Router VPN and Ethernet Services Command Reference.

SUMMARY STEPS

1. configure
2. l2vpn
3. bridge group bridge-group-name
4. bridge-domain bridge-domain-name
5. vfi {vfi-name}
6. vpn-id vpn-id
7. autodiscovery bgp
8. rd {as-number:nn \ ip-address:nn \ auto}
9. route-target {as-number:nn \ ip-address:nn \ export \ import}
10. route-target import {as-number:nn \ ip-address:nn}
11. route-target export {as-number:nn \ ip-address:nn}
12. signaling-protocol bgp
13. ve-id {number}
### 14. `ve-range {number}`

### 15. `commit`  

or  

`end`

---

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>l2vpn</code></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>bridge group bridge-group-name</code></td>
<td>Enters configuration mode for the named bridge group.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group metroA</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>bridge-domain bridge-domain-name</code></td>
<td>Enters configuration mode for the named bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain east</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> <code>vfi {vfi-name}</code></td>
<td>Enters virtual forwarding instance (VFI) configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi vfi-east</td>
<td></td>
</tr>
<tr>
<td><strong>Step 6</strong> <code>vpn-id vpn-id</code></td>
<td>Specifies the identifier for the VPLS service. The VPN ID has to be globally unique within a PE router. i.e., the same VPN ID cannot exist in multiple VFIs on the same PE router. In addition, a VFI can have only one VPN ID.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# vpn-id 100</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong> <code>autodiscovery bgp</code></td>
<td>Enters BGP autodiscovery configuration mode where all BGP autodiscovery parameters are configured.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# autodiscovery bgp</td>
<td>This command is not provisioned to BGP until at least the VPN ID and the signaling protocol is configured.</td>
</tr>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command/Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>`rd {as-number:nn</td>
<td>ip-address:nn</td>
</tr>
<tr>
<td>9</td>
<td>`route-target {as-number:nn</td>
<td>ip-address:nn}`</td>
</tr>
<tr>
<td>10</td>
<td>`route-target import {as-number:nn</td>
<td>ip-address:nn}`</td>
</tr>
<tr>
<td>11</td>
<td>`route-target export {as-number:nn</td>
<td>ip-address:nn}`</td>
</tr>
<tr>
<td>12</td>
<td><code>signaling-protocol bgp</code></td>
<td>Enables BGP signaling, and enters the BGP signaling configuration submode where BGP signaling parameters are configured. This command is not provisioned to BGP until VE ID and VE ID range is configured.</td>
</tr>
<tr>
<td>13</td>
<td><code>ve-id {number}</code></td>
<td>Specifies the local PE identifier for the VFI for VPLS configuration. The VE ID identifies a VFI within a VPLS service. This means that VFIs in the same VPLS service cannot share the same VE ID. The scope of the VE ID is only within a bridge domain. Therefore, VFIs in different bridge domains within a PE can use the same VE ID.</td>
</tr>
</tbody>
</table>
### Configuring VPLS with BGP Autodiscovery and LDP Signaling

Perform this task to configure BGP-based Autodiscovery and signaling:

#### SUMMARY STEPS

1. configure
2. l2vpn
3. router-id
4. bridge group bridge-group-name
5. bridge-domain bridge-domain-name
6. transport-mode vlan passthrough
7. vfi {vfi-name}
8. autodiscovery bgp
9. vpn-id vpn-id
10. rd {as-number:nn | ip-address:nn | auto}
11. route-target {as-number:nn | ip-address:nn | export | import}
12. `route-target import {as-number:nn | ip-address:nn}`
13. `route-target export {as-number:nn | ip-address:nn}`
14. `signaling-protocol ldp`
15. `vpls-id {as-number:nn | ip-address:nn}`
16. `commit`  
    or  
    `end`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1**  
  `configure` | Enters global configuration mode. |
| **Example:**  
  RP/0/RSP0/CPU0:router# configure | |
| **Step 2**  
  `l2vpn` | Enters L2VPN configuration mode. |
| **Example:**  
  RP/0/RSP0/CPU0:router(config)# l2vpn | |
| **Step 3**  
  `router-id ip-address` | Specifies a unique Layer 2 (L2) router ID for the provider edge (PE) router.  
  The router ID must be configured for LDP signaling, and is used as the L2 router ID in the BGP NLRI, SAII (local L2 Router ID) and TAII (remote L2 Router ID). Any arbitrary value in the IPv4 address format is acceptable.  
  **Note** Each PE must have a unique L2 router ID. This CLI is optional, as a PE automatically generates a L2 router ID using the LDP router ID. |
| **Example:**  
  RP/0/RSP0/CPU0:router(config-l2vpn)# router-id 1.1.1.1 | |
| **Step 4**  
  `bridge group bridge-group-name` | Enters configuration mode for the named bridge group. |
| **Example:**  
  RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group metroA | |
| **Step 5**  
  `bridge-domain bridge-domain-name` | Enters configuration mode for the named bridge domain. |
| **Example:**  
  RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain east | |
| **Step 6**  
  `transport-mode vlan passthrough` | Enables VC type 4 for BGP autodiscovery. |
| **Example:**  
  RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# transport-mode vlan passthrough | |
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>vfi (vfi-name)</td>
<td>Enters virtual forwarding instance (VFI) configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd)# vfi vfi-east</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>vpn-id vpn-id</td>
<td>Specifies the identifier for the VPLS service. The VPN ID has to be globally unique within a PE router, i.e., the same VPN ID cannot exist in multiple VFIs on the same PE router. In addition, a VFI can have only one VPN ID.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd-vfi)# vpn-id 100</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>autodiscovery bgp</td>
<td>Enters BGP autodiscovery configuration mode where all BGP autodiscovery parameters are configured. This command is not provisioned to BGP until at least the VPN ID and the signaling protocol is configured.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd-vfi)# autodiscovery bgp</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>rd {as-number:nn</td>
<td>ip-address:nn</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd-vfi-ad)# rd auto</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>route-target {as-number:nn</td>
<td>ip-address:nn}</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd-vfi-ad)# route-target 500:99</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>route-target import {as-number:nn</td>
<td>ip-address:nn}</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd-vfi-ad)# route-target import 200:20</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>route-target export {as-number:nn</td>
<td>ip-address:nn}</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd-vfi-ad)# route-target export 100:10</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>signaling-protocol ldp</td>
<td>Enables LDP signaling.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bd-bd-vfi-ad)# signaling-protocol ldp</td>
<td></td>
</tr>
</tbody>
</table>
How to Implement Multipoint Layer 2 Services

Command or Action          Purpose
---                        ---
Step 15 vpls-id \{as-number:nn\|ip-address:nn\} Specifies VPLS ID which identifies the VPLS domain during signaling.

This command is optional in all PEs that are in the same Autonomous System (share the same ASN) because a default VPLS ID is automatically generated using BGP’s ASN and the configured VPN ID (i.e., the default VPLS ID equals ASN:VPN-ID). If an ASN of 4 bytes is used, the lower two bytes of the ASN are used to build the VPLS ID. In case of InterAS, the VPLS ID must be explicitly configured. Only one VPLS ID can be configured per VFI, and the same VPLS ID cannot be used for multiple VFIs.

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-ad-sig)# vpls-id 10:20

Step 16 end or 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.

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-ad-sig)# end

or

RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-ad-sig)# commit

Configuring G.8032 Ethernet Ring Protection

To configure the G.8032 operation, separately configure:

  • An ERP instance to indicate:
    - which (sub)interface is used as the APS channel
    - which (sub)interface is monitored by CFM
    - whether the interface is an RPL link, and, if it is, the RPL node type
  
  • CFM with EFD to monitor the ring links

Note MEP for each monitor link needs to be configured with different Maintenance Association.
• The bridge domains to create the Layer 2 topology. The RAPS channel is configured in a dedicated management bridge domain separated from the data bridge domains.
• Behavior characteristics, that apply to ERP instance, if different from default values. This is optional.

This section provides information on:
• Configuring ERP Profile, page 315
• Configuring CFM MEP, page 316
• Configuring an ERP Instance, page 316
• Configuring ERP Parameters, page 320
• Configuring TCN Propagation, page 322

Configuring ERP Profile

Perform this task to configure Ethernet ring protection (ERP) profile.

SUMMARY STEPS

1. configure
   Example: RP/0/RSP0/CPU0:router# configure
   Enters global configuration mode.

2. ethernet ring g8032 profile profile-name
   Example: RP/0/RSP0/CPU0:router(config)# Ethernet ring g8032 profile p1
   Enables G.8032 ring mode, and enters G.8032 configuration submode.

3. timer {wtr | guard | hold-off} seconds
   Example: RP/0/RSP0/CPU0:router(config-g8032-ring-profile)# timer hold-off 5
   Specifies time interval (in seconds) for the guard, hold-off and wait-to-restore timers.

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> Ethernet ring g8032 profile profile-name</td>
<td>Enables G.8032 ring mode, and enters G.8032 configuration submode.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config)# Ethernet ring g8032 profile p1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> timer {wtr</td>
<td>guard</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-g8032-ring-profile)# timer hold-off 5</td>
<td></td>
</tr>
</tbody>
</table>
### How to Implement Multipoint Layer 2 Services

#### Configuring CFM MEP

For more information about Ethernet Connectivity Fault Management (CFM), refer to the *Configuring Ethernet OAM on the Cisco ASR 9000 Series Router* module in the Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide.

#### Configuring an ERP Instance

Perform this task to configure an ERP instance.

**SUMMARY STEPS**

1. `configure`
2. `l2vpn`
3. `bridge group bridge-group-name`
4. `bridge-domain aps-bridge-domain-name`
5. `interface type port0-interface-path-id.subinterface`
6. `interface type port1-interface-path-id.subinterface`
7. `bridge-domain data-bridge-domain-name`
8. `interface type interface-path-id.subinterface`

### Command or Action

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 4</strong> non-revertive</td>
<td>Specifies a non-revertive ring instance.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-g8032-ring-profile )# non-revertive</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> end or commit</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-g8032-ring-profile )# end or RP/0/RSP0/CPU0:router(config-g8032-ring-profile )# commit</td>
<td></td>
</tr>
</tbody>
</table>

*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.*
9. ethernet ring g8032 ring-name
10. instance number
11. description string
12. profile profile-name
13. rpl {port0 | port1} {owner | neighbor | next-neighbor}
14. inclusion-list vlan-ids vlan-id
15. aps-channel
16. level number
17. port0 interface type interface-path-id
18. port1 {interface type interface-path-id | bridge-domain bridge-domain-name | xconnect xconnect-name | none}
19. end
   or
commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> bridge group bridge-group-name</td>
<td>Creates a bridge group that can contain bridge domains, and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group csco RP/0/RSP0/CPU0:router(config-l2vpn-bg)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> bridge-domain bridge-domain-name</td>
<td>Establishes a bridge domain for R-APS channels, and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1 RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> interface type port0-interface-path-id.subinterface</td>
<td>Enters interface configuration mode and 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.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/0/0/0.1 RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#</td>
<td></td>
</tr>
</tbody>
</table>
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td><code>interface type port1-interface-path-id.subinterface</code></td>
<td>Enters interface configuration mode and 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.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
</tbody>
</table>
|      | RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
|      | interface GigabitEthernet 0/0/0.1 | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# | |
| 7    | `bridge-domain bridge-domain-name` | Establishes a bridge domain for data traffic, and enters L2VPN bridge group bridge domain configuration mode. |
|      | **Example:** | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-bg)#
|      | bridge-domain bd2 | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# | |
| 8    | `interface type interface-path-id.subinterface` | Enters interface configuration mode and 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. |
|      | **Example:** | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#
|      | interface GigabitEthernet 0/0/0/10 | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# | |
| 9    | `ethernet ring g8032 ring-name` | Enables G.8032 ring mode, and enters G.8032 configuration submode. |
|      | **Example:** | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn)#
|      | ethernet ring g8032 r1 | |
| 10   | `instance number` | Enters the Ethernet ring G.8032 instance configuration submode. |
|      | **Example:** | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-erp)#
|      | instance 1 | |
| 11   | `description string` | Specifies a string that serves as description for that instance. |
|      | **Example:** | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance)# description test | |
| 12   | `profile profile-name` | Specifies associated Ethernet ring G.8032 profile. |
|      | **Example:** | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance)#profile p1 | |
| 13   | `rpl (port0 | port1) (owner | neighbor | next-neighbor)` | Specifies one ring port on local node as RPL owner, neighbor or next-neighbor. |
|      | **Example:** | |
|      | RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance)#rpl port0 neighbor | |
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Step 14</th>
<th><strong>inclusion-list vlan-ids vlan-id</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Purpose</strong></td>
<td>Associates a set of VLAN IDs with the current instance.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance-aps)# inclusion-list vlan-ids e-g</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 15</th>
<th><strong>aps-channel</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Purpose</strong></td>
<td>Enters the Ethernet ring G.8032 instance aps-channel configuration submode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance-aps)# aps-channel</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 16</th>
<th><strong>level number</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Purpose</strong></td>
<td>Specifies the APS message level. The range is from 0 to 7.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance-aps)# level 5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 17</th>
<th><strong>port0 interface type interface-path-id</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Purpose</strong></td>
<td>Associates G.8032 APS channel interface to port0.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance-aps)# port0 interface GigabitEthernet 0/0/0/0.1</td>
</tr>
</tbody>
</table>

| Step 18 | **port1 (interface type interface-path-id | bridge-domain bridge-domain-name | xconnect xconnect-name | none)** |
| --- | --- |
| **Purpose** | Associates G.8032 APS channel interface to port1. |
| **Example:** | RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance-aps)# port1 interface GigabitEthernet 0/0/0/1.1 |

<table>
<thead>
<tr>
<th>Step 19</th>
<th><strong>end</strong> or <strong>commit</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Purpose</strong></td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance-aps)# end or RP/0/RSP0/CPU0:router(config-l2vpn-erp-instance-aps)# commit</td>
</tr>
</tbody>
</table>

- **When you issue the `end` command, the system prompts you to commit changes:**
  
  Uncommitted changes found, commit them before exiting(yes/no/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 ERP Parameters

Perform this task to configure ERP parameters.

SUMMARY STEPS

1. configure
2. l2vpn
3. ethernet ring g8032 ring-name
4. port0 interface type interface-path-id
5. monitor port0 interface type interface-path-id
6. exit
7. port1 (interface type interface-path-id | virtual | none)
8. monitor port1 interface type interface-path-id
9. exit
10. exclusion-list vlan-ids vlan-id
11. open-ring
12. end
   or
   commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> ethernet ring g8032 ring-name</td>
<td>Enables G.8032 ring mode, and enters G.8032 configuration submode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# ethernet ring g8032 r1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> port0 interface type interface-path-id</td>
<td>Enables G.8032 ERP for the specified port (ring port).</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-erp)# port0 interface GigabitEthernet 0/1/0/6</td>
<td></td>
</tr>
</tbody>
</table>
### Implementing Multipoint Layer 2 Services

#### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td><code>monitor port0 interface type interface-path-id</code></td>
<td>Specifies the port that is monitored to detect ring link failure per ring port. The monitored interface must be a sub-interface of the main interface.</td>
</tr>
<tr>
<td></td>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-l2vpn-erp-port0)# monitor port0 interface 0/1/0/2</code></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td><code>exit</code></td>
<td>Exits port0 configuration submode.</td>
</tr>
<tr>
<td></td>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-l2vpn-erp-port0)# exit</code></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>`port1 (interface type interface-path-id</td>
<td>virtual</td>
</tr>
<tr>
<td></td>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-l2vpn-erp)# port1 interface GigabitEthernet 0/1/0/8</code></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td><code>monitor port1 interface type interface-path-id</code></td>
<td>Specifies the port that is monitored to detect ring link failure per ring port. The monitored interface must be a sub-interface of the main interface.</td>
</tr>
<tr>
<td></td>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-l2vpn-erp-port1)# monitor port1 interface 0/1/0/3</code></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td><code>exit</code></td>
<td>Exits port1 configuration submode.</td>
</tr>
<tr>
<td></td>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-l2vpn-erp-port1)# exit</code></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td><code>exclusion-list vlan-ids vlan-id</code></td>
<td>Specifies a set of VLAN IDs that is not protected by Ethernet ring protection mechanism.</td>
</tr>
<tr>
<td></td>
<td>Example: <code>RP/0/RSP0/CPU0:router(config-l2vpn-erp)# exclusion-list vlan-ids a-d</code></td>
<td></td>
</tr>
</tbody>
</table>
Command or Action | Purpose
--- | ---
Step 11 open-ring | Specifies Ethernet ring G.8032 as open ring.
**Example:**
RP/0/RSP0/CPU0:router(config-l2vpn-erp)# open-ring
Step 12 end or commit | Saves configuration changes.
**Example:**
RP/0/RSP0/CPU0:router(config-l2vpn-erp)# end
or
RP/0/RSP0/CPU0:router(config-l2vpn-erp)# commit

**Configuring TCN Propagation**

Perform this task to configure topology change notification (TCN) propagation.

**SUMMARY STEPS**

1. configure
2. l2vpn
3. tcn-propagation
4. end or commit
DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> 12vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# 12vpn</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> tcn-propagation</td>
<td>Allows TCN propagation from minor ring to major ring and from MSTP to G.8032.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-12vpn)# tcn-propagation</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> end or commit</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-12vpn)# end or RP/0/RSP0/CPU0:router(config-12vpn)# commit</td>
<td></td>
</tr>
</tbody>
</table>

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 Flow Aware Transport Pseudowire

This section provides information on

- Enabling Load Balancing with ECMP and FAT PW for VPWS
- Enabling Load Balancing with ECMP and FAT PW for VPLS
Enabling Load Balancing with ECMP and FAT PW for VPWS

Perform this task to enable load balancing with ECMP and FAT PW for VPWS.

SUMMARY STEPS

1. configure
2. l2vpn
3. load-balancing flow \{src-dst-mac | src-dst-ip\}
4. pw-class \{name\}
5. encapsulation mpls
6. load-balancing flow-label \{both | code | receive | transmit\} [static]
7. exit
8. xconnect group group-name
9. p2p xconnect-name
10. interface type interface-path-id
11. neighbor A.B.C.D pw-id pseudowire-id
12. pw-class \{name\}
13. end
   or
   commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters the configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> load-balancing flow (src-dst-mac</td>
<td>src-dst-ip)</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# load-balancing flow src-dst-ip</td>
<td></td>
</tr>
</tbody>
</table>

- **src-dst-mac**—Uses source and destination MAC addresses for hashing.
- **src-dst-ip**—Uses source and destination IP addresses for hashing.

**Note** It is recommended to use the load-balancing flow command with the src-dst-ip keyword.
### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 4</td>
<td><code>pw-class (name)</code></td>
<td>Configures the pseudowire class template name to use for the pseudowire.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn)# pw-class path1</code></td>
<td></td>
</tr>
<tr>
<td>Step 5</td>
<td><code>encapsulation mpls</code></td>
<td>Configures the pseudowire encapsulation to MPLS.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls</code></td>
<td></td>
</tr>
<tr>
<td>Step 6</td>
<td>`load-balancing flow-label (both</td>
<td>code</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn-pwc-encap-mpls)# load-balancing flow-label both</code></td>
<td></td>
</tr>
<tr>
<td>Note</td>
<td>If the <code>static</code> keyword is not specified, end to end negotiation of the FAT PW is enabled.</td>
<td></td>
</tr>
<tr>
<td>Step 7</td>
<td><code>exit</code></td>
<td>Exits the pseudowire encapsulation submode and returns the router to the parent configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn-pwc-encap-mpls)#exit</code></td>
<td></td>
</tr>
<tr>
<td>Step 8</td>
<td><code>xconnect group group-name</code></td>
<td>Specifies the name of the cross-connect group.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn)# xconnect group grpl</code></td>
<td></td>
</tr>
<tr>
<td>Step 9</td>
<td><code>p2p xconnect-name</code></td>
<td>Specifies the name of the point-to-point cross-connect.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn-xc)# p2p vlan1</code></td>
<td></td>
</tr>
<tr>
<td>Step 10</td>
<td><code>interface type interface-path-id</code></td>
<td>Specifies the interface type and instance.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# interface GigabitEthernet0/0/0.1</code></td>
<td></td>
</tr>
<tr>
<td>Step 11</td>
<td><code>neighbor A.B.C.D pw-id pseudowire-id</code></td>
<td>Configures the pseudowire segment for the cross-connect. Use the A.B.C.D argument to specify the IP address of the cross-connect peer.</td>
</tr>
<tr>
<td>Example:</td>
<td><code>RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# neighbor 10.2.2.2 pw-id 2000</code></td>
<td>Note: A.B.C.D can be a recursive or non-recursive prefix.</td>
</tr>
</tbody>
</table>
Chapter Implementing Multipoint Layer 2 Services

How to Implement Multipoint Layer 2 Services

Enabling Load Balancing with ECMP and FAT PW for VPLS

Perform this task to enable load balancing with ECMP and FAT PW for VPLS.

SUMMARY STEPS

1. configure
2. l2vpn
3. load-balancing flow {src-dst-mac | src-dst-ip}
4. pw-class {class-name}
5. encapsulation mpls
6. load-balancing flow-label {both | code | receive | transmit} [static]
7. exit
8. bridge group bridge-group-name
9. bridge-domain bridge-domain-name
10. vfi {vfi-name}
11. autodiscovery bgp
12. signaling-protocol bgp
13. load-balancing flow-label {both | code | receive | transmit} [static]

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 12 pw-class class-name</td>
<td>Associates the pseudowire class with this pseudowire.</td>
</tr>
</tbody>
</table>

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)#
pw-class path1

Step 13 end
or
commit

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)#
end
or
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-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.
### How to Implement Multipoint Layer 2 Services

**Step 1**
- **Command or Action:** `configure`
  - **Purpose:** Enters the configuration mode.
  - **Example:**
    ```bash
    RP/0/RSP0/CPU0:router# configure
    ```

**Step 2**
- **Command or Action:** `l2vpn`
  - **Purpose:** Enters L2VPN configuration mode.
  - **Example:**
    ```bash
    RP/0/RSP0/CPU0:router(config)# l2vpn
    ```

**Step 3**
- **Command or Action:** `load-balancing flow {src-dst-mac | src-dst-ip}`
  - **Purpose:** Enables flow based load balancing.
    - **src-dst-mac:** Uses source and destination MAC addresses for hashing.
    - **src-dst-ip:** Uses source and destination IP addresses for hashing.
  - **Example:**
    ```bash
    RP/0/RSP0/CPU0:router(config-l2vpn)# load-balancing flow src-dst-ip
    ```

**Note:** It is recommended to use the `load-balancing flow` command with the `src-dst-ip` keyword.

**Step 4**
- **Command or Action:** `pw-class {class-name}`
  - **Purpose:** Associates the pseudowire class with this pseudowire.
  - **Example:**
    ```bash
    RP/0/RSP0/CPU0:router(config-l2vpn)# pw-class class1
    ```

**Step 5**
- **Command or Action:** `encapsulation mpls`
  - **Purpose:** Configures the pseudowire encapsulation to MPLS.
  - **Example:**
    ```bash
    RP/0/RSP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls
    ```

**Step 6**
- **Command or Action:** `load-balancing flow-label {both | code | receive | transmit} [static]`
  - **Purpose:** Enables load-balancing on ECMPs. Also, enables the imposition and disposition of flow labels for the pseudowire.
  - **Example:**
    ```bash
    RP/0/RSP0/CPU0:router(config-l2vpn-pwc-mpls)# load-balancing flow-label both
    ```

**Note:** If the `static` keyword is not specified, end to end negotiation of the FAT PW is enabled.

**Step 7**
- **Command or Action:** `exit`
  - **Purpose:** Exits the pseudowire encapsulation submode and returns the router to the parent configuration mode.
  - **Example:**
    ```bash
    RP/0/RSP0/CPU0:router(config-l2vpn-pwc-mpls)# exit
    ```
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>bridge group bridge-group-name</td>
<td>Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group group1</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>bridge-domain bridge-domain-name</td>
<td>Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg)#bridge-domain domain1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>vfi {vfi-name}</td>
<td>Enters virtual forwarding instance (VFI) configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#vfi my_vfi</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>autodiscovery bgp</td>
<td>Enters BGP autodiscovery configuration mode where all BGP autodiscovery parameters are configured.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# autodiscovery bgp</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>signaling-protocol bgp</td>
<td>Enables BGP signaling, and enters the BGP signaling configuration submode where BGP signaling parameters are configured.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-ad)# signaling-protocol bgp</td>
<td></td>
</tr>
</tbody>
</table>
Configuring Pseudowire Headend

The PWHE is created by configuring `pw-ether` interface. For the PWHE to be functional, the crossconnect has to be configured completely. Configuring other Layer 3 (L3) parameters, such as VRF and IP addresses, are optional for the PWHE to be functional. However, the L3 features are required for the Layer 3 services to be operational; that is, for PW L3 termination.

This section describes these topics:

- PWHE Configuration Restrictions, page 330
- Configuring PWHE Interfaces, page 330
- Configuring PWHE Crossconnect, page 332
- Configuring Generic Interface List, page 334
- Configuring the Source Address, page 336
- Configuring PWHE Interface Parameters, page 338

### Command or Action | Purpose
--- | ---
**Step 13**  
`load-balancing flow-label {both|code} receive|transmit} [static]`  
Example:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-ad-sig)# load-balancing flow-label both static  
Enables load-balancing on ECMPs. Also, enables the imposition and disposition of flow labels for the pseudowire.

**Step 14**  
end  
commit  
Example:  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-ad-sig)# end  
or  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-ad-sig)# 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.
PWHE Configuration Restrictions

These configuration restrictions are applicable for PWHE:

1. Only eight interface lists per peer are supported
2. Eight Layer 3 links per interface list are supported
3. VLAN ID (tag-impose) can be configured in crossconnects with pw-ether interfaces only
4. VLAN ID (tag-impose) can be configured under VC type 4 pw-ether interfaces only
5. Pseudowire redundancy, preferred path, local switching or L2TP are not supported for crossconnects configured with PWHE
6. Applications such as TE and LDP have checks for interface type and therefore do not allow PWHE to be configured.
7. Address family, CDP and MPLS configurations are not allowed on PWHE interfaces
8. Only eBGP and Static routes are supported

Configuring PWHE Interfaces

Perform this task to configure PWHE interfaces.

Summary Steps

1. configure
2. interface pw-ether id
3. attach generic-interface-list interface_list_name
4. end or commit

Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router# configure RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> interface pw-ether id</td>
<td>Configures the PWHE interface and enters the interface configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# interface pw-ether &lt;id&gt;</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step 3</th>
<th><code>attach generic-interface-list</code></th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>interface_list_name</code></td>
<td>Attaches the interface to a specified interface list.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router(config-if)# attach generic-interface-list interfacelist1
```

<table>
<thead>
<tr>
<th>Step 4</th>
<th><code>end</code> or <code>commit</code></th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Saves configuration changes.</td>
</tr>
</tbody>
</table>

#### 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 PWHE Crossconnect

Perform this task to configure PWHE crossconnects.

### Summary Steps

1. `configure`
2. `l2vpn`
3. `xconnect group group-name`
4. `p2p xconnect-name`
5. `interface pw-ether id`
6. `neighbor A.B.C.D pw-id value`
7. `pw-class class-name`
8. `end`
   or
   `commit`

### Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>
| **Example:**
  
  RP/0/RSP0/CPU0:router# `configure`
  
  RP/0/RSP0/CPU0:router(config)# | |
| **Step 2** `l2vpn` | Enters Layer 2 VPN configuration mode. |
| **Example:**
  
  RP/0/RSP0/CPU0:router(config)# `l2vpn` | |
| **Step 3** `xconnect group group-name` | Configures a cross-connect group name using a free-format 32-character string. |
| **Example:**
  
  RP/0/RSP0/CPU0:router(config-l2vpn)# `xconnect group MS-PW1` | |
| **Step 4** `p2p xconnect-name` | Enters P2P configuration submode. |
| **Example:**
  
  RP/0/RSP0/CPU0:router(config-l2vpn-xc)# `p2p ms-pw1` | |
| **Step 5** `interface pw-ether id` | Configures the PWHE interface. |
| **Example:**
  
  RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# `interface pw-ether 100` | |
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 6**
neighbor A.B.C.D pw-id value  
**Example:**
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p)# neighbor 10.165.200.25 pw-id 100  | Configures a pseudowire for a cross-connect.  
The IP address is that of the corresponding PE node.  
The **pw-id** must match the **pw-id** of the PE node. |
| **Step 7**
pw-class class-name  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-xc-p2p-pw)# pw-class dynamic_mpls | Enters pseudowire class submode, allowing you to define a pseudowire class template.  
**Note**  
The pseudowire class should be defined under l2vpn for VC4 and VC5 as follows:  
```plaintext  
pw-class vc_type_4  
encapsulation mpls  
transport-mode vlan  
!  
pw-class vc_type_5  
encapsulation mpls  
transport-mode ethernet  
!  
```
| **Step 8**
end  
or
commit  
**Example:**  
RP/0/RSP0/CPU0:router(config-if)# end  
or  
RP/0/RSP0/CPU0:router(config-if)# 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 Generic Interface List

Perform this task to configure a generic interface list.

### Summary Steps

1. `configure`
2. `generic-interface-list list-name`
3. `interface type interface-path-id`
4. `interface type interface-path-id`
5. `end`
   or
   `commit`

### Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>generic-interface-list list-name</code></td>
<td>Configures a generic interface list.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
</tr>
<tr>
<td>generic-interface-list list1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>interface type interface-path-id</code></td>
<td>Configures the specified interface.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-if-list)#</td>
<td></td>
</tr>
<tr>
<td>interface Bundle-Ether 100</td>
<td></td>
</tr>
</tbody>
</table>
## How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 4</strong></td>
<td></td>
</tr>
<tr>
<td><code>interface type interface-path-id</code></td>
<td>Configures the specified interface.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-if-list)# interface Bundle-Ether 200</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td></td>
</tr>
<tr>
<td><code>end</code> or <code>commit</code></td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-if-list)# end</td>
<td></td>
</tr>
<tr>
<td>or</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-if-list)# commit</td>
<td></td>
</tr>
</tbody>
</table>

- 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 the Source Address

Perform this task to configure the local source address.

Summary Steps

1. configure
2. l2vpn
3. pw-class class-name
4. encapsulation mpls
5. ipv4 source source-address
6. end
   or
   commit

Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router#</td>
<td>System prompt: configure</td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> l2vpn</td>
<td>Enters Layer 2 VPN configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)#</td>
<td>System prompt: l2vpn</td>
</tr>
<tr>
<td><strong>Step 3</strong> pw-class class-name</td>
<td>Enters pseudowire class submode, allowing you to define a pseudowire class template.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)#</td>
<td>System prompt: pw-class class1</td>
</tr>
<tr>
<td><strong>Step 4</strong> encapsulation mpls</td>
<td>Configures the pseudowire encapsulation to MPLS.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-pwc)#</td>
<td>System prompt: encapsulation mpls</td>
</tr>
</tbody>
</table>
### Command or Action | Purpose
--- | ---
**Step 5**  
`ipv4 source source-address`  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-pwc-mp ls)# ipv4 source 10.1.1.1  
| Sets the local source IPv4 address.  

**Step 6**  
`end`  
_or_  
`commit`  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-pwc-mp ls)# end  
or  
RP/0/RSP0/CPU0:router(config-l2vpn-pwc-mp ls)# 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 PWHE Interface Parameters

Perform this task to configure PWHE interface parameters.

### Summary Steps

1. configure
2. interface pw-ether *id*
3. attach generic-interface-list *interface_list_name*
4. l2overhead *bytes*
5. load-interval *seconds*
6. dampening *decay-life*
7. logging events link-status
8. mac-address *MAC address*
9. mtu *interface_MTU*
10. bandwidth *kbps*
11. end
   or
   commit

### Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>
| **Example:** RP/0/RSP0/CPU0:router# configure  
RP/0/RSP0/CPU0:router(config)# | |
| **Step 2** interface pw-ether *id* | Configures the PWHE interface and enters the interface configuration mode. |
| **Example:** RP/0/RSP0/CPU0:router(config)# interface pw-ether <id> | |
| **Step 3** attach generic-interface-list *interface_list_name* | Attaches the interface to a specified interface list. |
| **Example:** RP/0/RSP0/CPU0:router(config-if)# attach generic-interface-list interface_list1 | |
| **Step 4** l2overhead *bytes* | Sets layer 2 overhead size. |
| **Example:** RP/0/RSP0/CPU0:router(config-if)#l2overhead 20 | |
## How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| Step 5 | `load-interval seconds` | Specifies interval, in seconds, for load calculation for an interface. The number of seconds:  
- Can be set to 0 [0 disables load calculation]  
- If not 0, interval must be specified in multiples of 30 between 30 and 600. |
| Example: | `RP/0/RSP0/CPU0:router(config-if)#load-interval 90` |
| Step 6 | `dampening decay-life` | Configures state dampening on the given interface (in minutes). |
| Example: | `RP/0/RSP0/CPU0:router(config-if)#dampening 10` |
| Step 7 | `logging events link-status` | Configures per interface logging. |
| Example: | `RP/0/RSP0/CPU0:router(config-if)#logging events link-status` |
| Step 8 | `mac-address MAC address` | Sets the MAC address (xxxx.xxxxxx) on an interface. |
| Example: | `RP/0/RSP0/CPU0:router(config-if)#mac-address aaaa.bbbb.cccc` |
| Step 9 | `mtu interface_MTU` | Sets the MTU on an interface. |
| Example: | `RP/0/RSP0/CPU0:router(config-if)#mtu 128` |
### How to Implement Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 10</strong></td>
<td>Configures the bandwidth. The range is between 0 to 4294967295 kbps.</td>
</tr>
<tr>
<td><code>bandwidth kbps</code></td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# bandwidth 200</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Step 11</strong></th>
<th>Saves configuration changes.</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>end</code> or <code>commit</code></td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# end</td>
</tr>
<tr>
<td>or</td>
<td>RP/0/RSP0/CPU0:router(config-if)# commit</td>
</tr>
</tbody>
</table>

- 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 L2VPN over GRE

Perform these tasks to configure L2VPN over GRE.

SUMMARY STEPS

1. configure
2. interface type interface-path-id
3. l2transport
4. exit
5. interface loopback instance
6. ipv4 address ip-address
7. exit
8. interface loopback instance
9. ipv4 address ip-address
10. router ospf process-name
11. area area-id
12. interface type interface-path-id
13. interface tunnel-ip number
14. exit
15. interface tunnel-ip number
16. ipv4 address ipv4-address mask
17. tunnel source type path-id
18. tunnel destination ip-address
19. end
20. l2vpn
21. bridge group bridge-group-name
22. bridge-domain bridge-domain-name
23. interface type interface-path-id
24. neighbor \{A.B.C.D\} \{pw-id value\}
25. mpls ldp
26. router-id \{router-id\}
27. interface tunnel-ip number
28. end
   or
commit
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><code>interface type interface-path-id</code></td>
<td>Enters interface configuration mode and configures an interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# interface TenGigE0/1/0/12</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><code>l2transport</code></td>
<td>Enables Layer 2 transport on the selected interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# l2transport</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>exit</code></td>
<td>Exits the current configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# exit</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td><code>interface loopback instance</code></td>
<td>Enters interface configuration mode and names the new loopback interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# interface Loopback0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td><code>ipv4 address ip-address</code></td>
<td>Assigns an IP address and subnet mask to the virtual loopback interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# ipv4 address 100.100.100.100 255.255.255</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td><code>exit</code></td>
<td>Exits the current configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# exit</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td><code>interface loopback instance</code></td>
<td>Enters interface configuration mode and names the new loopback interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# interface Loopback1</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td><code>ipv4 address ip-address</code></td>
<td>Assigns an IP address and subnet mask to the virtual loopback interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# ipv4 address 10.0.1.1 255.255.255.255</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>------</td>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td>10</td>
<td>router ospf process-name</td>
<td>Enables OSPF routing for the specified routing process and places the router in router configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router# router ospf 1</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>area area-id</td>
<td>Enters area configuration mode and configures an area for the OSPF process.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router# area 0</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>interface loopback instance</td>
<td>Enters interface configuration mode and names the new loopback interface.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router# interface Loopback0</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>interface tunnel-ip number</td>
<td>Enters tunnel interface configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router# interface tunnel-ip1</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>exit</td>
<td>Exits the current configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router# exit</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>interface tunnel-ip number</td>
<td>Enters tunnel interface configuration mode. &lt;br&gt;• number is the number associated with the tunnel interface.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router(config)# interface tunnel-ip1</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>ipv4 address ipv4-address subnet-mask</td>
<td>Specifies the IPv4 address and subnet mask for the interface. &lt;br&gt;• ipv4-address specifies the IP address of the interface. &lt;br&gt;• subnet-mask specifies the subnet mask of the interface.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router(config-if)# ipv4 address 12.0.0.1 255.255.255.0</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>tunnel source type path-id</td>
<td>Specifies the source of the tunnel interface.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router(config-if)# tunnel source Loopback1</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>tunnel destination ip-address</td>
<td>Defines the tunnel destination.</td>
</tr>
<tr>
<td></td>
<td>Example: &lt;br&gt;RP/0/RSP0/CPU0:router(config-if)# tunnel destination 100.100.100.20</td>
<td></td>
</tr>
</tbody>
</table>
### How to Implement Multipoint Layer 2 Services

#### Command or Action

<table>
<thead>
<tr>
<th>Step 19</th>
<th>end</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# end</td>
</tr>
</tbody>
</table>

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)?
  - 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.

<table>
<thead>
<tr>
<th>Step 20</th>
<th>l2vpn</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# l2vpn</td>
</tr>
</tbody>
</table>

Enters L2VPN configuration mode.

<table>
<thead>
<tr>
<th>Step 21</th>
<th>bridge group bridge-group-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# bridge group access-pw</td>
</tr>
</tbody>
</table>

Creates a bridge group that can contain bridge domains, and then assigns network interfaces to the bridge domain.

<table>
<thead>
<tr>
<th>Step 22</th>
<th>bridge-domain bridge-domain-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# bridge-domain test</td>
</tr>
</tbody>
</table>

Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

<table>
<thead>
<tr>
<th>Step 23</th>
<th>interface type interface-path-id</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# interface TenGigE0/1/0/12</td>
</tr>
</tbody>
</table>

Enters interface configuration mode and 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.

<table>
<thead>
<tr>
<th>Step 24</th>
<th>neighbor (A.B.C.D) (pw-id value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# neighbor 125.125.125.125 pw-id 100</td>
</tr>
</tbody>
</table>

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.

  **Note**: A.B.C.D can be a recursive or non-recursive prefix.

- Use the pw-id keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.
### Configuring a GRE Tunnel as Preferred Path for Pseudowire

Perform this task to configure a GRE tunnel as the preferred path for pseudowires.

**SUMMARY STEPS**

1. `configure`
2. `l2vpn`
3. `pw-class {name}`
4. `encapsulation mpls`

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| Step 25
`mpls ldp` | Enables MPLS LDP configuration mode. |
| Example:
RP/0/RSP0/CPU0:router# mpls ldp | |
| Step 26
`router-id (router-id)` | Configures a router ID for the OSPF process. |
| Example:
RP/0/RSP0/CPU0:router# router-id 100.100.100.100 | Note We recommend using a stable IP address as the router ID. |
| Step 27
`interface tunnel-ip number` | Enters tunnel interface configuration mode. |
| Example:
RP/0/RSP0/CPU0:router# interface tunnel-ip1 | Note The number argument refers to the number associated with the tunnel interface |
| Step 28
`end` or
`commit` | Saves configuration changes. |
| Example:
RP/0/RSP0/CPU0:router(config-if)# end or
RP/0/RSP0/CPU0:router(config-if)# commit | |

- 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.
5. `preferred-path {interface} {tunnel-ip value | tunnel-te value | tunnel-tp value} [fallback disable]`

6. `end`
   
or
   `commit`

## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters the configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>l2vpn</code></td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# l2vpn</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>pw-class {name}</code></td>
<td>Configures the pseudowire class name.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn)# pw-class gre</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>encapsulation mpls</code></td>
<td>Configures the pseudowire encapsulation to MPLS.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls</td>
<td></td>
</tr>
</tbody>
</table>
### Step 5

**Command or Action:**

```
preferred-path [interface] (tunnel-ip value | tunnel-te value | tunnel-tp value) [fallback disable]
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-pwc-encap-mpls)# preferred-path interface tunnel-ip 1 fallback disable
```

**Purpose:**

Configures preferred path tunnel settings. If the fallback disable configuration is used and once the TE/TP tunnel is configured as the preferred path goes down, the corresponding pseudowire can also go down.

**Note:** Ensure that fallback is supported.

### Step 6

**Command or Action:**

```
end
or
commit
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-pwc-encap-mpls)# end
```

or

```
RP/0/RSP0/CPU0:router(config-l2vpn-pwc-encap-mpls-if)# commit
```

**Purpose:**

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.
Configuration Examples for Multipoint Layer 2 Services

This section includes these configuration examples:

- Virtual Private LAN Services Configuration for Provider Edge-to-Provider Edge: Example, page 348
- Virtual Private LAN Services Configuration for Provider Edge-to-Customer Edge: Example, page 349
- Displaying MAC Address Withdrawal Fields: Example, page 351
- Split Horizon Group: Example, page 352
- Blocking Unknown Unicast Flooding: Example, page 353
- Disabling MAC Flush: Examples, page 353
- Configuring VPLS with BGP Autodiscovery and Signaling: Example, page 361
- Bridging on IOS XR Trunk Interfaces: Example, page 354
- Bridging on Ethernet Flow Points: Example, page 358
- Changing the Flood Optimization Mode: Example, page 360
- Configuring VPLS with BGP Autodiscovery and Signaling: Example, page 361
- Configuring Dynamic ARP Inspection: Example, page 366
- Configuring IP Source Guard: Example, page 368
- Configuring G.8032 Ethernet Ring Protection: Example, page 369
- Configuring Flow Aware Transport Pseudowire: Example, page 373
- Configuring Pseudowire Headend: Example, page 374
- Configuring L2VPN over GRE: Example, page 376

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.

This configuration example shows how to configure PE 1:

```
configure
l2vpn
   bridge group 1
   bridge-domain PE1-VPLS-A
   GigabitEthernet0/0/0/1
   vfi 1
   neighbor 10.2.2.2 pw-id 1
   neighbor 10.3.3.3 pw-id 1
   !
   interface loopback 0
   ipv4 address 10.1.1.1 255.255.255.255
```

This configuration example shows how to configure PE 2:

```
configure
```
l2vpn
  bridge group 1
    bridge-domain PE2-VPLS-A
      interface GigabitEthernet0/0/0/1
        vfi 1
          neighbor 10.1.1.1 pw-id 1
          neighbor 10.3.3.3 pw-id 1
          
        !
    !
  !
interface loopback 0
  ipv4 address 10.2.2.2 255.255.255.25

This configuration example shows how to configure PE 3:

configure
l2vpn
  bridge group 1
    bridge-domain PE3-VPLS-A
      interface GigabitEthernet0/0/0/1
        vfi 1
          neighbor 10.1.1.1 pw-id 1
          neighbor 10.2.2.2 pw-id 1
          
        !
    !
  !
interface loopback 0
  ipv4 address 10.3.3.3 255.255.255.25

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

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

configure
interface GigabitEthernet0/0/0/1
  l2transport---AC interface
    no ipv4 address
    no ipv4 directed-broadcast
    negotiation auto
    no cdp enable

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

configure
interface GigabitEthernet0/0
  l2transport
    no ipv4 address
    no ipv4 directed-broadcast
    negotiation auto
no cdp enable
Displaying MAC Address Withdrawal Fields: Example

This sample output shows the MAC address withdrawal fields:

```
RP/0/RSP0/CPU0# 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 10.1.1.1, 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
```
Split Horizon Group: Example

This example configures interfaces for Layer 2 transport, adds them to a bridge domain, and assigns them to split horizon groups.

```
RP/0/RSP0/CPU0:router(config)#l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)#bridge group examples
RP/0/RSP0/CPU0:router(config-l2vpn)#bridge-domain all_three
RP/0/RSP0/CPU0:router(config-l2vpn-bd)#interface GigabitEthernet 0/0/0/0.99
RP/0/RSP0/CPU0:router(config-l2vpn-bd-ac)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bd-ac)#split-horizon group
RP/0/RSP0/CPU0:router(config-l2vpn-bd-ac)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bd-adj)#neighbor 192.168.99.1 pw-id 1
RP/0/RSP0/CPU0:router(config-l2vpn-bd-pw)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bd-adj)#neighbor 192.168.99.9 pw-id 1
RP/0/RSP0/CPU0:router(config-l2vpn-bd-adj)#split-horizon group
RP/0/RSP0/CPU0:router(config-l2vpn-bd-adj)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bd-adj)#vfi abc
RP/0/RSP0/CPU0:router(config-l2vpn-bd-adj)#neighbor 192.168.99.17 pw-id 1
RP/0/RSP0/CPU0:router(config-l2vpn-bd-adj)#show
```

According to this example, the Split Horizon group assignments for bridge domain `all_three` are:

<table>
<thead>
<tr>
<th>Bridge Port/Pseudowire</th>
<th>Split Horizon Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>bridge port: gig0/0/0/0.99</td>
<td>0</td>
</tr>
<tr>
<td>bridge port: gig0/0/0/0.101</td>
<td>2</td>
</tr>
<tr>
<td>PW: 192.168.99.1 pw-id 1</td>
<td>0</td>
</tr>
<tr>
<td>PW: 192.168.99.9 pw-id 1</td>
<td>2</td>
</tr>
<tr>
<td>PW: 192.168.99.17 pw-id 1</td>
<td>1</td>
</tr>
</tbody>
</table>
Blocking Unknown Unicast Flooding: Example

Unknown-unicast flooding can be blocked at these levels:

- bridge domain
- bridge port (attachment circuit (AC))
- access pseudowire (PW)

This example shows how to block unknown-unicast flooding at the bridge domain level:

```
configure
l2vpn
   bridge-group group1
   bridge-domain domain1
   flooding unknown-unicast disable
end
```

This example shows how to block unknown-unicast flooding at the bridge port level:

```
configure
l2vpn
   bridge-group group1
   bridge-domain domain1
   interface GigabitEthernet 0/1/0/1
   flooding unknown-unicast disable
end
```

This example shows how to block unknown-unicast flooding at the access pseudowire level:

```
configure
l2vpn
   bridge-group group1
   bridge-domain domain1
   neighbor 10.1.1.1 pw-id 1000
   flooding unknown-unicast disable
end
```

Disabling MAC Flush: Examples

You can disable the MAC flush at these levels:

- bridge domain
- bridge port (attachment circuit (AC))
- access pseudowire (PW)

This example shows how to disable the MAC flush at the bridge domain level:

```
configure
l2vpn
   bridge-group group1
   bridge-domain domain1
   mac
   port-down flush disable
end
```
This example shows how to disable the MAC flush at the bridge port level:

```
configure
l2vpn
   bridge-group group1
   bridge-domain domain1
   interface GigabitEthernet 0/1/0/1
   mac
      port-down flush disable
end
```

This example shows how to disable the MAC flush at the access pseudowire level:

```
configure
l2vpn
   bridge-group group1
   bridge-domain domain1
   neighbor 10.1.1.1 pw-id 1000
   mac
      port-down flush disable
end
```

### Bridging on IOS XR Trunk Interfaces: Example

This example shows how to configure a Cisco ASR 9000 Series Router as a simple L2 switch.

**Important Notes:**

Create a bridge domain that has four attachment circuits (AC). Each AC is an IOS XR trunk interface (i.e. not a subinterface/EFP).

- This example assumes that the running config is empty, and that all the components are created.
- This example provides all the necessary steps to configure the Cisco ASR 9000 Series Router to perform switching between the interfaces. However, the commands to prepare the interfaces such as no shut, negotiation auto, etc., have been excluded.
- The bridge domain is in a no shut state, immediately after being created.
- Only trunk (i.e. main) interfaces are used in this example.
- The trunk interfaces are capable of handling tagged (i.e. IEEE 802.1Q) or untagged (i.e. no VLAN header) frames.
- The bridge domain learns, floods, and forwards based on MAC address. This functionality works for frames regardless of tag configuration.
- The bridge domain entity spans all the line cards of the system. It is not necessary to place all the bridge domain ACs on a single LC. This applies to any bridge domain configuration.
- The show bundle and the show l2vpn bridge-domain commands are used to verify that the router was configured as expected, and that the commands show the status of the new configurations.
- The ACs in this example use interfaces that are in the admin down state.
# Configuration Example

```
RP/0/RSP0/CPU0:router(config)#interface Bundle-ether10
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#interface GigabitEthernet0/2/0/5
RP/0/RSP0/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP0/CPU0:router(config-if)#interface GigabitEthernet0/2/0/6
RP/0/RSP0/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP0/CPU0:router(config-if)#interface GigabitEthernet0/2/0/0
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#interface GigabitEthernet0/2/0/1
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#interface TenGigE0/1/0/2
RP/0/RSP0/CPU0:router(config-if)#l2transport
RP/0/RSP0/CPU0:router(config-if-l2)#l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)#bridge group examples
RP/0/RSP0/CPU0:router(config-l2vpn-bg)#bridge-domain test-switch
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface Bundle-ether10
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface GigabitEthernet0/2/0/0
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface GigabitEthernet0/2/0/1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)#interface TenGigE0/1/0/2
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#commit
RP/0/RSP0/CPU0:Jul 26 10:48:21.320 EDT: config[65751]: %MGBL-CONFIG-6-DB_COMMIT : Configuration committed by user 'lab'. Use 'show configuration commit changes 1000000973' to view the changes.
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)#end
RP/0/RSP0/CPU0:Jul 26 10:48:21.342 EDT: config[65751]: %MGBL-SYS-5-CONFIG_I : Configured from console by lab
RP/0/RSP0/CPU0:router#show bundle Bundle-ether10

Bundle-Ether10
Status: Down
Local links <active/standby/configured>: 0 / 0 / 2
Local bandwidth <effective/available>: 0 (0) kbps
MAC address (source): 0024.f71e.22eb (Chassis pool)
Minimum active links / bandwidth: 1 / 1 kbps
Maximum active links: 64
Wait while timer: 2000 ms
LACP: Operational
Flap suppression timer: Off
mLACP: Not configured
IPv4 BFD: Not configured

Port Device State Port ID B/W, kbps
-------------- --------------- ----------- -------------- ----------
Gi0/2/0/5 Local Configured 0x8000, 0x0001 1000000
Gi0/2/0/6 Local Configured 0x8000, 0x0002 1000000

RP/0/RSP0/CPU0:router# show 12vpm bridge-domain group examples
Bridge group: examples, bridge-domain: test-switch, id: 2000, state: up, ShgId: 0, MSTi: 0
Aging: 300 s, MAC limit: 4000, Action: none, Notification: syslog
Filter MAC addresses: 0
ACs: 4 (1 up), VFIIs: 0, PWS: 0 (0 up), PBBs: 0 (0 up)
List of ACs:
BE10, state: down, Static MAC addresses: 0
Gi0/2/0/0, state: up, Static MAC addresses: 0
Gi0/2/0/1, state: down, Static MAC addresses: 0
```
Te0/5/0/1, state: down, Static MAC addresses: 0
List of Access PWs:
List of VFI:s:
RP/0/RSP0/CPU0:router#

This table lists the configuration steps (actions) and the corresponding purpose for this example:

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Step 2 interface Bundle-ether10</td>
<td>Creates a new bundle trunk interface.</td>
</tr>
<tr>
<td>Step 3 l2transport</td>
<td>Changes Bundle-ether10 from an L3 interface to an L2 interface.</td>
</tr>
<tr>
<td>Step 4 interface GigabitEthernet0/2/0/5</td>
<td>Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/5.</td>
</tr>
<tr>
<td>Step 5 bundle id 10 mode active</td>
<td>Establishes GigabitEthernet0/2/0/5 as a member of Bundle-ether10. The <strong>mode active</strong> keywords specify LACP protocol.</td>
</tr>
<tr>
<td>Step 6 interface GigabitEthernet0/2/0/6</td>
<td>Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/6.</td>
</tr>
<tr>
<td>Step 7 bundle id 10 mode active</td>
<td>Establishes GigabitEthernet0/2/0/6 as a member of Bundle-ether10. The <strong>mode active</strong> keywords specify LACP protocol.</td>
</tr>
<tr>
<td>Step 8 interface GigabitEthernet0/2/0/0</td>
<td>Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/0.</td>
</tr>
<tr>
<td>Step 9 l2transport</td>
<td>Change GigabitEthernet0/2/0/0 from an L3 interface to an L2 interface.</td>
</tr>
<tr>
<td>Step 10 interface GigabitEthernet0/2/0/1</td>
<td>Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/2/0/1.</td>
</tr>
<tr>
<td>Step 11 l2transport</td>
<td>Change GigabitEthernet0/2/0/1 from an L3 interface to an L2 interface.</td>
</tr>
<tr>
<td>Step 12 interface TenGigE0/1/0/2</td>
<td>Enters interface configuration mode. Changes configuration mode to act on TenGigE0/1/0/2.</td>
</tr>
<tr>
<td>Step 13 l2transport</td>
<td>Changes TenGigE0/1/0/2 from an L3 interface to an L2 interface.</td>
</tr>
<tr>
<td>Step 14 l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Step 15 bridge group examples</td>
<td>Creates the bridge group <strong>examples</strong>.</td>
</tr>
<tr>
<td>Step 16 bridge-domain test-switch</td>
<td>Creates the bridge domain <strong>test-switch</strong>, that is a member of bridge group <strong>examples</strong>.</td>
</tr>
<tr>
<td>Step 17 interface Bundle-ether10</td>
<td>Establishes Bundle-ether10 as an AC of bridge domain test-switch.</td>
</tr>
<tr>
<td>Step 18 exit</td>
<td>Exits bridge domain AC configuration submode, allowing next AC to be configured.</td>
</tr>
</tbody>
</table>
### Configuration Examples for Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td><code>interface GigabitEthernet0/2/0/0</code></td>
<td>Establishes GigabitEthernet0/2/0/0 as an AC of bridge domain test-switch.</td>
</tr>
<tr>
<td>20</td>
<td><code>exit</code></td>
<td>Exits bridge domain AC configuration submode, allowing next AC to be configured.</td>
</tr>
<tr>
<td>21</td>
<td><code>interface GigabitEthernet0/2/0/1</code></td>
<td>Establishes GigabitEthernet0/2/0/1 as an AC of bridge domain test-switch.</td>
</tr>
<tr>
<td>22</td>
<td><code>exit</code></td>
<td>Exits bridge domain AC configuration submode, allowing next AC to be configured.</td>
</tr>
<tr>
<td>23</td>
<td><code>interface TenGigE0/1/0/2</code></td>
<td>Establishes interface TenGigE0/1/0/2 as an AC of bridge domain test-switch.</td>
</tr>
<tr>
<td>24</td>
<td><code>end</code> or <code>commit</code></td>
<td>Saves configuration changes.</td>
</tr>
</tbody>
</table>

- 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.
Bridging on Ethernet Flow Points: Example

This example shows how to configure a Cisco ASR 9000 Series Router to perform Layer 2 switching on traffic that passes through Ethernet Flow Points (EFPs). EFP traffic typically has one or more VLAN headers. Although both IOS XR trunks and IOS XR EFPs can be combined as attachment circuits in bridge domains, this example uses EFPs exclusively.

**Important Notes:**
- An EFP is a Layer 2 subinterface. It is always created under a trunk interface. The trunk interface must exist before the EFP is created.
- In an empty configuration, the bundle interface trunk does not exist, but the physical trunk interfaces are automatically configured when a line card is inserted. Therefore, only the bundle trunk is created.
- In this example the subinterface number and the VLAN IDs are identical, but this is out of convenience, and is not a necessity. They do not need to be the same values.
- The bridge domain test-efp has three attachment circuits (ACs). All the ACs are EFPs.
- Only frames with a VLAN ID of 999 enter the EFPs. This ensures that all the traffic in this bridge domain has the same VLAN encapsulation.
- The ACs in this example use interfaces that are in the admin down state, or interfaces for which no line card has been inserted (**unresolved** state). Bridge domains that use nonexistent interfaces as ACs are legal, and the commit for such configurations does not fail. In this case, the status of the bridge domain shows **unresolved** until you configure the missing interface.

**Configuration Example**

```
RP/0/RSP1/CPU0:router#configure
RP/0/RSP1/CPU0:router(config)#interface Bundle-ether10
RP/0/RSP1/CPU0:router(config-if)#interface Bundle-ether10.999 l2transport
RP/0/RSP1/CPU0:router(config-subif)#encapsulation dot1q 999
RP/0/RSP1/CPU0:router(config-subif)#interface GigabitEthernet0/6/0/5
RP/0/RSP1/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP1/CPU0:router(config-if)#interface GigabitEthernet0/6/0/6
RP/0/RSP1/CPU0:router(config-if)#bundle id 10 mode active
RP/0/RSP1/CPU0:router(config-if)#interface GigabitEthernet0/6/0/7.999 l2transport
RP/0/RSP1/CPU0:router(config-subif)#encapsulation dot1q 999
RP/0/RSP1/CPU0:router(config-subif)#interface TenGigE0/1/0/2.999 l2transport
RP/0/RSP1/CPU0:router(config-subif)#encapsulation dot1q 999
RP/0/RSP1/CPU0:router(config-subif)#l2vpn
RP/0/RSP1/CPU0:router(config-l2vpn)#bridge group examples
RP/0/RSP1/CPU0:router(config-l2vpn-bg)#bridge-domain test-efp
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd)#interface Bundle-ether10.999
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd)#interface GigabitEthernet0/6/0/5
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#exit
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#commit
RP/0/RSP1/CPU0:router(config-l2vpn-bg-bd-ac)#end
RP/0/RSP1/CPU0:router#show l2vpn bridge group examples
Fri Jul 23 21:56:34.473 UTC Bridge group: examples, bridge-domain: test-efp, id: 0, state: up, ShgId: 0, MSTi: 0
Aging: 300 s, MAC limit: 4000, Action: none, Notification: syslog
Filter MAC addresses: 0
ACs: 3 (0 up), VFIs: 0, PWS: 0 (0 up), PBBs: 0 (0 up)
List of ACs:
  BE10.999, state: down, Static MAC addresses: 0
```
Gi0/6/0/7.999, state: unresolved, Static MAC addresses: 0
Te0/1/0/2.999, state: down, Static MAC addresses: 0
List of Access PWs:
List of VFIs:
RP/0/RSP1/CPU0:router#

This table lists the configuration steps (actions) and the corresponding purpose for this example:

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Step 2 interface Bundle-ether10</td>
<td>Creates a new bundle trunk interface.</td>
</tr>
<tr>
<td>Step 3 interface Bundle-ether10.999 l2transport</td>
<td>Creates an EFP under the new bundle trunk.</td>
</tr>
<tr>
<td>Step 4 encapsulation dot1q 999</td>
<td>Assigns VLAN ID of 999 to this EFP.</td>
</tr>
<tr>
<td>Step 5 interface GigabitEthernet0/6/0/5</td>
<td>Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/6/0/5.</td>
</tr>
<tr>
<td>Step 6 bundle id 10 mode active</td>
<td>Establishes GigabitEthernet0/6/0/5 as a member of Bundle-ether10. The mode active keywords specify LACP protocol.</td>
</tr>
<tr>
<td>Step 7 interface GigabitEthernet0/6/0/6</td>
<td>Enters interface configuration mode. Changes configuration mode to act on GigabitEthernet0/6/0/6.</td>
</tr>
<tr>
<td>Step 8 bundle id 10 mode active</td>
<td>Establishes GigabitEthernet0/6/0/6 as a member of Bundle-ether10. The mode active keywords specify LACP protocol.</td>
</tr>
<tr>
<td>Step 9 interface GigabitEthernet0/6/0/7.999 l2transport</td>
<td>Creates an EFP under GigabitEthernet0/6/0/7.</td>
</tr>
<tr>
<td>Step 10 encapsulation dot1q 999</td>
<td>Assigns VLAN ID of 999 to this EFP.</td>
</tr>
<tr>
<td>Step 11 interface TenGigE0/1/0/2.999 l2transport</td>
<td>Creates an EFP under TenGigE0/1/0/2.</td>
</tr>
<tr>
<td>Step 12 encapsulation dot1q 999</td>
<td>Assigns VLAN ID of 999 to this EFP.</td>
</tr>
<tr>
<td>Step 13 l2vpn</td>
<td>Enters L2VPN configuration mode.</td>
</tr>
<tr>
<td>Step 14 bridge group examples</td>
<td>Creates the bridge group named examples.</td>
</tr>
<tr>
<td>Step 15 bridge-domain test-efp</td>
<td>Creates the bridge domain named test-efp, that is a member of bridge group examples.</td>
</tr>
<tr>
<td>Step 16 interface Bundle-ether10.999</td>
<td>Establishes Bundle-ether10.999 as an AC of the bridge domain named test-efp.</td>
</tr>
<tr>
<td>Step 17 exit</td>
<td>Exits bridge domain AC configuration submode, allowing next AC to be configured.</td>
</tr>
<tr>
<td>Step 18 interface GigabitEthernet0/6/0/7.999</td>
<td>Establishes GigabitEthernet0/6/0/7.999 as an AC of the bridge domain named test-efp.</td>
</tr>
<tr>
<td>Step 19 exit</td>
<td>Exits bridge domain AC configuration submode, allowing next AC to be configured.</td>
</tr>
</tbody>
</table>
Chapter | Implementing Multipoint Layer 2 Services

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 20</td>
<td>Interface TenGigE0/1/0/2.999 Establishes interface TenGigE0/1/0/2.999 as an AC of bridge domain named test-efp.</td>
</tr>
<tr>
<td>Step 21</td>
<td>end or commit Saves configuration changes.</td>
</tr>
<tr>
<td></td>
<td>● When you issue the end command, the system prompts you to commit changes:</td>
</tr>
<tr>
<td></td>
<td>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</td>
</tr>
<tr>
<td></td>
<td>– Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>– Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>– Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>● Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.</td>
</tr>
</tbody>
</table>

**Changing the Flood Optimization Mode: Example**

This example shows how to change the default flood optimization mode under a bridge domain:

```
config
l2vpn
  bridge group MyGroup
  bridge-domain MyDomain
  flood mode convergence-optimized
```
Configuring VPLS with BGP Autodiscovery and Signaling: Example

This section contains these configuration examples for configuring the BGP autodiscovery and signaling feature:

- LDP and BGP Configuration
- Minimum L2VPN Configuration for BGP Autodiscovery with BGP Signaling
- VPLS with BGP Autodiscovery and BGP Signaling
- Minimum Configuration for BGP Autodiscovery with LDP Signaling
- VPLS with BGP Autodiscovery and LDP Signaling

LDP and BGP Configuration

Figure 15 illustrates an example of LDP and BGP configuration.

```
Configuration at PE1:
interface Loopback0
   ipv4 address 1.1.1.100 255.255.255.255
!
interface Loopback1
   ipv4 address 1.1.1.10 255.255.255.255
!
mls ldp
   router-id 1.1.1.1
   interface GigabitEthernet0/1/0/0
!
routing bgp 120
   address-family l2vpn vpls-vpws
!
neighbor 2.2.2.20
   remote-as 120
   update-source Loopback1
   address-family l2vpn vpls-vpws
   signaling bgp disable

Configuration at PE2:
interface Loopback0
   ipv4 address 2.2.2.200 255.255.255.255
!
interface Loopback1
   ipv4 address 2.2.2.20 255.255.255.255
!
mls ldp
   router-id 2.2.2.2
   interface GigabitEthernet0/1/0/0
!
routing bgp 120
```
Minimum L2VPN Configuration for BGP Autodiscovery with BGP Signaling

This example illustrates the minimum L2VPN configuration required for BGP Autodiscovery with BGP Signaling, where any parameter that has a default value is not configured.

```
(config)# l2vpn
(config-l2vpn)# bridge group {bridge group name}
(config-l2vpn-bg)# bridge-domain {bridge domain name}
(config-l2vpn-bg-md)# vfi (vfi name)
(config-l2vpn-bg-md-vfi)# autodiscovery bgp
(config-l2vpn-bg-md-vfi-ad)# vpn-id 10
(config-l2vpn-bg-md-vfi-ad)# rd auto
(config-l2vpn-bg-md-vfi-ad)# route-target 1.1.1.1:100
(config-l2vpn-bg-md-vfi-ad-sig)# signaling-protocol bgp
(config-l2vpn-bg-md-vfi-ad-sig)# ve-id 1
(config-l2vpn-bg-md-vfi-ad-sig)# commit
```

VPLS with BGP Autodiscovery and BGP Signaling

Figure 16 illustrates an example of configuring VPLS with BGP autodiscovery (AD) and BGP Signaling.

```
VPLS with BGP autodiscovery and BGP signaling

Configuration at PE1:

l2vpn
  bridge group gr1
  bridge-domain bd1
  interface GigabitEthernet0/1/0/1.1
  vfi vf1
  ! AD independent VFI attributes
  vpn-id 100
  ! Auto-discovery attributes
  autodiscovery bgp
  rd auto
  route-target 2.2.2.2:100
  ! Signaling attributes
  signaling-protocol bgp
  ve-id 3
```
Configuration at PE2:

```
l2vpn
bridge group gr1
bridge-domain bd1
  interface GigabitEthernet0/1/0/2.1
vfi vf1
! AD independent VFI attributes
vpn-id 100
! Auto-discovery attributes
autodiscovery bgp
rd auto
route-target 2.2.2.2:100
! Signaling attributes
signaling-protocol bgp
ve-id 5
```

This is an example of NLRI for VPLS with BGP AD and signaling:

```
Discovery Attributes

NLRI sent at PE1:
Length = 19
Router Distinguisher = 3.3.3.3:32770
VE ID = 3
VE Block Offset = 1
VE Block Size = 10
Label Base = 16015

NLRI sent at PE2:
Length = 19
Router Distinguisher = 1.1.1.1:32775
VE ID = 5
VE Block Offset = 1
VE Block Size = 10
Label Base = 16120
```

Minimum Configuration for BGP Autodiscovery with LDP Signaling

This example illustrates the minimum L2VPN configuration required for BGP Autodiscovery with LDP Signaling, where any parameter that has a default value is not configured.

```
(config)# l2vpn
(config-l2vpn)# bridge group {bridge group name}
(config-l2vpn-bg)# bridge-domain {bridge domain name}
(config-l2vpn-bg-bd)# vfi {vfi name}
(config-l2vpn-bg-bd-vfi)# autodiscovery bgp
(config-l2vpn-bg-bd-vfi-ad)# vpn-id 10
(config-l2vpn-bg-bd-vfi-ad)# rd auto
(config-l2vpn-bg-bd-vfi-ad)# route-target 1.1.1.1:100
(config-l2vpn-bg-bd-vfi-ad)# commit
```
VPLS with BGP Autodiscovery and LDP Signaling

Figure 17 illustrates an example of configuring VPLS with BGP autodiscovery (AD) and LDP Signaling.

Figure 17  VPLS with BGP autodiscovery and LDP signaling

Configuration at PE1:

l2vpn
  router-id 10.10.10.10
  bridge group bg1
  bridge-domain bd1
  vfi vf1
  vpn-id 100
  autodiscovery bgp
  rd 1:100
  router-target 12:12

Configuration at PE2:

l2vpn
  router-id 20.20.20.20
  bridge group bg1
  bridge-domain bd1
  vfi vf1
  vpn-id 100
  autodiscovery bgp
  rd 2:200
  router-target 12:12
  signaling-protocol ldp
  vpls-id 120:100

Discovery and Signaling Attributes

Configuration at PE1:

LDP Router ID - 1.1.1.1
BGP Router ID - 1.1.1.100
Peer Address - 1.1.1.10
L2VPN Router ID - 10.10.10.10
Route Distinguisher - 1:100
**Common Configuration between PE1 and PE2:**

ASN - 120  
VPN ID - 100  
VPLS ID - 120:100  
Route Target - 12:12

**Configuration at PE2:**

LDP Router ID - 2.2.2.2  
BGP Router ID - 2.2.2.200  
Peer Address - 2.2.2.20  
L2VPN Router ID - 20.20.20.20  
Route Distinguisher - 2:200

**Discovery Attributes**

**NLRI sent at PE1:**

Source Address - 1.1.1.10  
Destination Address - 2.2.2.20  
Length - 14  
Route Distinguisher - 1:100  
L2VPN Router ID - 10.10.10.10  
VPLS ID - 120:100  
Route Target - 12:12

**NLRI sent at PE2:**

Source Address - 2.2.2.20  
Destination Address - 1.1.1.10  
Length - 14  
Route Distinguisher - 2:200  
L2VPN Router ID - 20.20.20.20  
VPLS ID - 120:100  
Route Target - 12:12

**Enabling VC type 4 for BGP Autodiscovery**

This example shows how to configure virtual connection type 4 in VPLS with BGP autodiscovery:

```
l2vpn
bridge group bg1
bridge-domain bd1
  transport-mode vlan passthrough
  interface GigabitEthernet0/0/0/1
  !
neighbor 2.2.2.2 pw-id 1
  !
vfi vf1
  vpn-id 100
  autodiscovery bgp
  rd auto
  route-target 1:1
  signaling-protocol ldp
  !
  !
  !
```
Configuring Dynamic ARP Inspection: Example

This example shows how to configure basic dynamic ARP inspection under a bridge domain:

```config
l2vpn
  bridge group MyGroup
  bridge-domain MyDomain
  dynamic-arp-inspection logging
```

This example shows how to configure basic dynamic ARP inspection under a bridge port:

```config
l2vpn
  bridge group MyGroup
  bridge-domain MyDomain
  interface gigabitEthernet 0/1/0/0.1
  dynamic-arp-inspection logging
```
This example shows how to configure optional dynamic ARP inspection under a bridge domain:

```
l2vpn
  bridge group SECURE
  bridge-domain SECURE-DAI
  dynamic-arp-inspection
      logging
      address-validation
          src-mac
          dst-mac
      ipv4
```

This example shows how to configure optional dynamic ARP inspection under a bridge port:

```
l2vpn
  bridge group SECURE
  bridge-domain SECURE-DAI
  interface GigabitEthernet0/0/0/1.10
      dynamic-arp-inspection
          logging
          address-validation
              src-mac
              dst-mac
          ipv4
```

This example shows the output of the `show l2vpn bridge-domain bd-name SECURE-DAI detail` command:

```
#show l2vpn bridge-domain bd-name SECURE-DAI detail
Bridge group: SECURE, bridge-domain: SECURE-DAI, id: 2, state: up,
...
Dynamic ARP Inspection: enabled, Logging: enabled
Dynamic ARP Inspection Address Validation:
  IPv4 verification: enabled
  Source MAC verification: enabled
  Destination MAC verification: enabled
...
List of ACs:
  AC: GigabitEthernet0/0/0/1.10, state is up
...
Dynamic ARP Inspection: enabled, Logging: enabled
Dynamic ARP Inspection Address Validation:
  IPv4 verification: enabled
  Source MAC verification: enabled
  Destination MAC verification: enabled
  IP Source Guard: enabled, Logging: enabled
...
Dynamic ARP inspection drop counters:
  packets: 1000, bytes: 64000
```

This example shows the output of the `show l2vpn forwarding interface interface-name detail location location-name` command:

```
#show l2vpn forwarding interface g0/0/0/1.10 det location 0/0/CPU0
Local interface: GigabitEthernet0/0/0/1.10, Xconnect id: 0x40001, Status: up
...
Dynamic ARP Inspection: enabled, Logging: enabled
Dynamic ARP Inspection Address Validation:
  IPv4 verification: enabled
  Source MAC verification: enabled
  Destination MAC verification: enabled
  IP Source Guard: enabled, Logging: enabled
```
This example shows the logging display:

```
LC/0/0/CPU0:Jun 16 13:28:28.697 : l2fib[188]: %L2-L2FIB-5-SECURITY_DAI_VIOLATION_AC:
Dynamic ARP inspection in AC GigabitEthernet0_0_0_7.100 detected violated packet - source
MAC: 0000.0000.0065, destination MAC: 0000.0040.0000, sender MAC: 0000.0000.0064, target
MAC: 0000.0000.0000, sender IP: 5.6.6.6, target IP: 130.10.3.2
```

```
LC/0/5/CPU0:Jun 16 13:28:38.716 : l2fib[188]: %L2-L2FIB-5-SECURITY_DAI_VIOLATION_AC:
Dynamic ARP inspection in AC Bundle-Ether100.103 detected violated packet - source MAC:
0000.0000.0067, destination MAC: 0000.2300.0000, sender MAC: 0000.7800.0034, target MAC:
0000.0000.0000, sender IP: 130.2.5.1, target IP: 50.5.1.25
```

---

### Configuring IP Source Guard: Example

This example shows how to configure basic IP source guard under a bridge domain:

```
config
l2vpn
  bridge group MyGroup
  bridge-domain MyDomain
  ip-source-guard logging
```

This example shows how to configure basic IP source guard under a bridge port:

```
config
l2vpn
  bridge group MyGroup
  bridge-domain MyDomain
  interface gigabitEthernet 0/1/0/0.1
  ip-source-guard logging
```

This example shows how to configure optional IP source guard under a bridge domain:

```
l2vpn
  bridge group SECURE
    bridge-domain SECURE-IPSG
    ip-source-guard
      logging
```

This example shows how to configure optional IP source guard under a bridge port:

```
l2vpn
  bridge group SECURE
    bridge-domain SECURE-IPSG
    interface GigabitEthernet0/0/0/1.10
    ip-source-guard
    logging
```

This example shows the output of the `show l2vpn bridge-domain bd-name ipsg-name detail` command:

```
# show l2vpn bridge-domain bd-name SECURE-IPSG detail
Bridge group: SECURE, bridge-domain: SECURE-IPSG, id: 2, state: up,
  IP Source Guard: enabled, Logging: enabled
  List of ACs:
    AC: GigabitEthernet0/0/0/1.10, state is up
      IP Source Guard: enabled, Logging: enabled
      IP source guard drop counters:
```

---

**Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide**

1368
This example shows the output of the `show l2vpn forwarding interface interface-name detail location location-name` command:

```
# show l2vpn forwarding interface g0/0/0/1.10 detail location 0/0/CPU0
Local interface: GigabitEthernet0/0/0/1.10, Xconnect id: 0x40001, Status: up

---
IP Source Guard: enabled, Logging: enabled
```

This example shows the logging display:

LC/0/0/CPU0:Jun 16 13:32:25.334 : l2fib[188]: %L2-L2FIB-5-SECURITY_IPSG_VIOLATION_AC : IP source guard in AC GigabitEthernet0_0_0_7.1001 detected violated packet - source MAC: 0000.0000.0200, destination MAC: 0000.0003.0000, source IP: 130.0.0.1, destination IP: 125.34.2.5

LC/0/5/CPU0:Jun 16 13:33:25.530 : l2fib[188]: %L2-L2FIB-5-SECURITY_IPSG_VIOLATION_AC : IP source guard in AC Bundle-Ether100.100 detected violated packet - source MAC: 0000.0000.0064, destination MAC: 0000.0040.0000, source IP: 14.5.1.3, destination IP: 45.1.1.10

---

### Configuring G.8032 Ethernet Ring Protection: Example

This sample configuration illustrates the elements that a complete G.8032 configuration includes:

```
# Configure the ERP profile characteristics if ERP instance behaviors are non-default.
ethernet ring g8032 profile ERP-profile
  timer wtr 60
  timer guard 100
  timer hold-off 1
  non-revertive

# Configure CFM MEPs and configure to monitor the ring links.
ethernet cfm
  domain domain1
    service link1 down-meps
    continuity-check interval 100ms
    efd
    mep crosscheck
      mep-id 2
  domain domain2
    service link2 down-meps
    continuity-check interval 100ms
    efd protection-switching
    mep crosscheck
      mep id 2

Interface Gig 0/0/0/0
  ethernet cfm mep domain domain1 service link1 mep-id 1
Interface Gig 1/1/0/0
  ethernet cfm mep domain domain2 service link2 mep-id 1

# Configure the ERP instance under L2VPN
l2vpn
  ethernet ring g8032 RingA
    port0 interface g0/0/0/0
    port1 interface g0/1/0/0
    instance 1
```
description BD2-ring
profile ERP-profile
rpl port0 owner
vlan-ids 10-100
aps channel
level 3
port0 interface g0/0/0/0.1
port1 interface g1/1/0/0.1

# Set up the bridge domains
bridge group ABC
bridge-domain BD2
  interface Gig 0/0/0/0.2
  interface Gig 0/1/0/0.2
  interface Gig 0/2/0/0.2

bridge-domain BD2-APS
  interface Gig 0/0/0/0.1
  interface Gig 1/1/0/0.1

# EFPs configuration
interface Gig 0/0/0/0.1 l2transport
  encapsulation dot1q 5
interface Gig 1/1/0/0.1 l2transport
  encapsulation dot1q 5
interface g 0/0/0/0.2 l2transport
  encapsulation dot1q 10-100
interface g 0/1/0/0.2 l2transport
  encapsulation dot1q 10-100
interface g 0/2/0/0.2 l2transport
  encapsulation dot1q 10-100

Configuring Interconnection Node: Example

This example shows you how to configure an interconnection node. Figure 18 illustrates an open ring scenario.
The minimum configuration required for configuring G.8032 at Router C (Open ring – Router C):

```
interface <ifname1.1> l2transport
    encapsulation dot1q X1
interface <ifname1.10> l2transport
    encapsulation dot1q Y1
interface <ifname2.10> l2transport
    encapsulation dot1q Y1
interface <ifname3.10> l2transport
    encapsulation dot1q Y1
l2vpn
ethernet ring g8032 <ring-name>
    port0 interface <main port ifname1>
    port1 interface none #? This router is connected to an interconnection node
    open-ring #? Mandatory when a router is part of an open-ring
    instance <1-2>
    inclusion-list vlan-ids X1-Y1
    aps-channel
        Port0 interface <ifname1.1>
        Port1 none #? This router is connected to an interconnection node
        bridge group bg1
    bridge-domain bd-aps#? APS-channel has its own bridge domain
        <ifname1.1> #? There is only one APS-channel at the interconnection node
    bridge-domain bd-traffic #? Data traffic has its own bridge domain
        <ifname1.10>
        <ifname2.10>
        <ifname3.10>
```

**Configuring the Node of an Open Ring: Example**

This example shows you how to configure the node part of an open ring. Figure 19 illustrates an open ring scenario.
The minimum configuration required for configuring G.8032 at the node of the open ring (node part of the open ring at router F):

```
interface <ifname1.1> l2transport
   encapsulation dot1q X1
interface <ifname1.10> l2transport
   encapsulation dot1q Y1
interface <ifname2.1> l2transport
   encapsulation dot1q X1
interface <ifname2.10> l2transport
   encapsulation dot1q Y1
l2vpn
   ethernet ring g8032 <ring-name>
      port0 interface <main port ifname1>
      port1 interface <main port ifname2>
      open-ring #? Mandatory when a router is part of an open-ring instance <1-2>
         inclusion-list vlan-ids X1-Y1
      rpl port1 owner #? This node is RPL owner and <main port ifname2> is blocked
                     aps-channel
         port0 interface <ifname1.1>
         port1 interface <ifname2.1>
      bridge group bg1
      bridge-domain bd-aps #? APS-channel has its own bridge domain
         <ifname1.1>
         <ifname2.1>
      bridge-domain bd-traffic #? Data traffic has its own bridge domain
         <ifname1.10>
         <ifname2.10>
```
**Configuring Flow Aware Transport Pseudowire: Example**

This sample configuration shows how to enable load balancing with FAT PW for VPWS.

```
l2vpn
pw-class class1
  encapsulation mpls
  load-balancing flow-label transmit
!
!
 pw-class class2
  encapsulation mpls
  load-balancing flow-label both
!

xconnect group group1
  p2p p1
    interface GigabitEthernet 0/0/0/0.1
    neighbor 1.1.1.1 pw-id 1
      pw-class class1
    !
    !
```

This sample configuration shows how to enable load balancing with FAT PW for VPLS.

```
Note
For VPLS, the configuration at the bridge-domain level is applied to all PWs (access and VFI PWs). Pseudowire classes are defined to override the configuration for manual PWs.
```

```
l2vpn
pw-class class1
  encapsulation mpls
  load-balancing flow-label both

bridge group group1
  bridge-domain domain1
    vfi vfi2-auto-bgp
      autodiscovery bgp
        signaling-protocol bgp
          load-balancing flow-label both static
        !
        !
        !
    bridge-domain domain2
    vfi vfi2-auto-ldp
      autodiscovery bgp
        signaling-protocol ldp
          load-balancing flow-label both static
        !
        !
    !
```
Configuring Pseudowire Headend: Example

This example shows how to configure pseudowire headend. Consider the topology in Figure 20.

**Figure 20  Pseudowire Headend Example**

There are multiple CEs connected to A-PE (each CE is connected by one link). There are two P routers between A-PE and S-PE in the access network. The S-PE is connected using two links to P1. These links L1 and L2 (on two different line cards on P1 and S-PE), e.g. Gig0/1/0/0 and Gig0/2/0/0 respectively.

The S-PE is connected by two links to P2, links L3 and L4 (on two different line cards on P2 and S-PE), e.g. Gig0/1/0/1 and Gig0/2/0/1 respectively. For each CE-APE link, a xconnect (AC-PW) is configured on the A-PE. A-PE uses router-id 100.100.100.100 for routing and PW signaling. Two router-ids on S-PE used for PW signaling, e.g. 111.111.111.111 and 112.112.112.112 (for rx pin-down), 110.110.110.110 is the router-id for routing.

**CE Configuration**

Consider two CEs that are connected through Ge0/3/0/0 (CE1 and A-PE) and Ge0/3/0/1 (CE2 and A-PE).

**CE1**

```
interface Gig0/3/0/0
ipv4 address 10.1.1.1/24
router static
  address-family ipv4 unicast
  110.110.110.110 Gig0/3/0/0
  A.B.C.D/N 110.110.110.110
```

**CE2**

```
interface Gig0/3/0/1
ipv4 address 10.1.2.1/24
router static
  address-family ipv4 unicast
  110.110.110.110 Gig0/3/0/1
  A.B.C.D/N 110.110.110.110
```
A-PE Configuration

At A-PE we have 1 xconnect for each CE connection. Here we give the configuration for the 2 CE connections above: both connections are L2 links which are in xconnects. Each xconnect has a PW destined to S-PE but we use a different neighbor address depending of where we want to pin-down the PW: [L1, L4] or [L2, L3]

interface Gig0/3/0/0
l2transport
interface Gig0/3/0/1
l2transport

l2vpn
xconnect group pwhe
  p2p pwhe_spe_1
    interface Gig0/3/0/0
    neighbor 111.111.111.111 pw-id 1
  p2p pwhe_spe_2
    interface Gig0/3/0/1
    neighbor 112.112.112.112 pw-id 2

P Router Configuration

We need static routes on P routers for rx pindown on S-PE, i.e. to force PWs destined to a specific address to be transported over certain links.

P1

router static
  address-family ipv4 unicast
    111.111.111.111 Gig0/1/0/0
    112.112.112.112 Gig0/2/0/0

P2

router static
  address-family ipv4 unicast
    111.111.111.111 Gig0/2/0/1
    112.112.112.112 Gig0/1/0/1

S-PE Configuration

At S-PE we have 2 PW-HE interfaces (1 for each PW) and each uses a different interface list for tx pin-down (has to match the static config at P routers for rx pin-down). Each PW-HE has its PW going to A-PE (pw-id has to match what's at A-PE).

generic-interface-list il1
  interface gig0/1/0/0
  interface gig0/2/0/0

 generic-interface-list il2
   interface gig0/1/0/1
   interface gig0/2/0/1

interface pw-ether1
  ipv4 address 10.1.1.2/24
  attach generic-interface-list il1
interface pw-ether2
  ipv4 address 10.1.2.2/24
  attach generic-interface-list il2

l2vpn
xconnect group pwhe
   p2p pwhe1
     interface pw-ether1
     neighbor 100.100.100.100 pw-id 1
   p2p pwhe2
     interface pw-ether2
     neighbor 100.100.100.100 pw-id 2

## Configuring L2VPN over GRE: Example

Configure the PW core interfaces under IGP and ensure that the loopback is reachable. Configure the tunnel source such that the tunnel is the current loopback as well as destination of the peer PE loopback. Now, configure the GRE tunnel in IGP (OSPF or ISIS), and also under mpls ldp and ensure that the LDP neighbor is established between the PEs for the PW. This ensures that the PW is Up through the tunnel.

**Configuration on PE1:**

```
router ospf 1
router-id 1.1.1.1
   area 0
     interface Loopback0
     interface TenGigE0/0/0/1
router ospf 2
router-id 200.200.200.200
   area 0
     interface Loopback1000
     interface tunnel-ip1
mpls ldp
router-id 200.200.200.200
interface tunnel-ip1
```

**Configuration on PE2:**

```
router ospf 1
router-id 3.3.3.3
   area 0
     interface Loopback0
     interface TenGigE0/2/0/3
router ospf 2
router-id 201.201.201.201
   area 0
     interface Loopback1000
     interface tunnel-ip1
! mpls ldp
router-id 201.201.201.201
interface tunnel-ip1
```

## Configuring a GRE Tunnel as the Preferred Path for Pseudowire: Example

This example shows how to configure a GRE tunnel as the preferred path for a pseudowire.

```
l2vpn
   pw-class gre
   encapsulation mpls
   preferred-path interface tunnel-ip 1 fallback disable
```
Additional References

For additional information related to implementing VPLS, refer to these:

Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XR L2VPN commands</td>
<td>Point to Point Layer 2 Services Commands module in the Cisco ASR 9000 Series Aggregation Services Router VPN and Ethernet Services Command Reference</td>
</tr>
<tr>
<td>MPLS VPLS-related commands</td>
<td>Multipoint Layer 2 Services Commands module in the Cisco ASR 9000 Series Aggregation Services Router VPN and Ethernet Services Command Reference</td>
</tr>
<tr>
<td>Getting started material</td>
<td>Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide</td>
</tr>
<tr>
<td>Traffic storm control on VPLS bridges</td>
<td>Traffic Storm Control under VPLS Bridges on Cisco ASR 9000 Series Routers module in the Cisco ASR 9000 Series Aggregation Services Router System Security Configuration Guide</td>
</tr>
<tr>
<td>Layer 2 multicast on VPLS bridges</td>
<td>Layer 2 Multicast Using IGMP Snooping module in the Cisco ASR 9000 Series Aggregation Services Router Multicast Configuration Guide</td>
</tr>
</tbody>
</table>

Standards

<table>
<thead>
<tr>
<th>Standards1</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>draft-ietf-l2vpn-vpls-ldp-09</td>
<td>Virtual Private LAN Services Using LDP</td>
</tr>
</tbody>
</table>

1. Not all supported standards are listed.

MIBs

<table>
<thead>
<tr>
<th>MIBs</th>
<th>MIBs Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at this URL and choose a platform under the Cisco Access Products menu: <a href="http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml">http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml</a></td>
</tr>
</tbody>
</table>
### RFCs

<table>
<thead>
<tr>
<th>RFC</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 4447</td>
<td><em>Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)</em>, April 2006</td>
</tr>
<tr>
<td>RFC 4448</td>
<td><em>Encapsulation Methods for Transport of Ethernet over MPLS Networks</em>, April 2006</td>
</tr>
<tr>
<td>RFC 4762</td>
<td><em>Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling</em></td>
</tr>
</tbody>
</table>

### Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td><a href="http://www.cisco.com/techsupport">http://www.cisco.com/techsupport</a></td>
</tr>
</tbody>
</table>