MPLS Layer 2 VPNs Configuration Guide, Cisco IOS Everest 16.5

First Published: 2017-08-28

Last Modified: 2017-08-28

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

CHAPTER 1

Read Me First

Important Information about Cisco IOS XE 16

Effective Cisco IOS XE Release 3.7.0E for Catalyst Switching and Cisco IOS XE Release 3.17S (for Access and Edge Routing) the two releases evolve (merge) into a single version of converged release—the Cisco IOS XE 16—providing one release covering the extensive range of access and edge products in the Switching and Routing portfolio.

Feature Information

Use Cisco Feature Navigator to find information about feature support, platform support, and Cisco software image support. An account on Cisco.com is not required.

Related References

- Cisco IOS Command References, All Releases

Obtaining Documentation and Submitting a Service Request

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

CHAPTER 2

L2VPN Protocol-Based CLIs

The L2VPN Protocol-Based CLIs feature provides a set of processes and an improved infrastructure for developing and delivering Cisco IOS software on various Cisco platforms. This feature introduces new commands and modifies or replaces existing commands to achieve a consistent functionality across Cisco platforms and provide cross-Operating System (OS) support.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About L2VPN Protocol-Based CLIs

Overview of L2VPN Protocol-Based CLIs

The L2VPN Protocol-Based CLIs feature introduces new commands and modifies or replaces existing commands to achieve a consistent functionality across Cisco platforms and provide cross-Operating System (OS) support.

Note The new, updated, and replacement commands are available in Cisco IOS XE Release 3.7S and Cisco IOS Release 15.3(1)S. However, the legacy commands that are being replaced will be deprecated in later releases.

Benefits of L2VPN Protocol-Based CLIs

The L2VPN Protocol-Based CLIs feature provides the following benefits:

- Consistent user experience across different operating systems.
- Consistent configuration for all Layer 2 VPN (L2VPN) scenarios.
- Enhanced functionality that is achieved by configuring pseudowires as virtual interfaces and monitoring the pseudowires as physical ports.
- Feature configuration such as quality of service (QoS) service policies on individual pseudowires .
- Redundant pseudowire configuration that is independent of the primary pseudowire to provide enhanced high availability.

These benefits are achieved through the following enhancements:

- New service contexts can be created for point-to-point and multipoint Layer 2 services by using the new L2VPN cross connect and L2VPN virtual forwarding interface (VFI) contexts.
  - The L2VPN cross connect context is used for configuring point-to-point pseudowires, pseudowire stitching, and local switching (hair pinning). Ethernet interfaces , Ethernet Flow Points (EFP), ATM interfaces and WAN interfaces (PPP,HDLC,Serial), and pseudowire interfaces can be defined as members of an L2VPN cross connect context.
  - The L2VPN VFI context instantiates Virtual Private LAN Services (VPLS) VFI for multipoint scenarios. Pseudowires can be defined as members of an L2VPN VFI context.
  - Bridge domains are used for multipoint scenarios. EFPs, pseudowires, or VFIs can be configured as members of a bridge domain. Pseudowires can be configured as member of a VFI. The VFI can be configured as a member of a .
- New port contexts can be created (dynamically or manually) for pseudowires by using the pseudowire interface.
- Pseudowire customization can be achieved using interface templates and pseudowire interfaces that are applied to L2VPN context members. Pseudowire customizations include following features:
  - Encapsulation type
  - Control word
  - Maximum Transmission Unit (MTU)
  - Pseudowire signaling type
  - Tunnel selection
- Interworking and redundancy group service attributes can be configured under the L2VPN service context. The redundancy groups are configured independently from the primary pseudowire, which helps achieve zero traffic interruptions while adding, modifying, or deleting backup pseudowires.

L2VPN Protocol-Based CLI Changes

The following commands are introduced in Cisco IOS XE Release 3.7S, Cisco IOS Release 15.3(1)S, and Cisco IOS Release 15.4(1)S:

- **debug l2vpn pseudowire**
- **l2vpn**
- **l2vpn pseudowire static-oam class**
- **monitor event-trace l2vpn**
- **show interface pseudowire**
- **show l2vpn service**
- **shutdown (MPLS)**
- **vc**

The following commands are modified in Cisco IOS XE Release 3.7S and Cisco IOS Release 15.3(1)S:

- **auto-route-target**
- **bridge-domain parameterized vlan**
- **debug condition xconnect fib**
- **debug condition xconnect interface**
- **debug condition xconnect peer**
- **debug condition xconnect segment**
- **description**
- **encapsulation (MPLS)**
- **forward permit l2protocol all**
- **interworking**
- **l2vpn subscriber authorization group**
- **l2vpn xconnect context**
- **load-balance flow**
- **monitor event-trace ac**
- **monitor event-trace atom**
- **monitor event-trace l2tp**
- **monitor peer bfd**
- **mtu**
- **preferred-path**
- **remote circuit id**

- **rd (VPLS)**
- **route-target (VPLS)**
- **sequencing**
- **status**
- **status admin-down disconnect**
- **status control-plane route-watch**
- **status decoupled**
- **status peer topology dual-homed**
- **status protocol notification static**
- **status redundancy**
- **switching tlv**
- **tlv**
- **tlv template**
- **vccv**
- **vccv bfd status signaling**
- **vccv bfd template**
- **vpls-id**
- **vpn id (MPLS)**

The table below lists the legacy commands that will be replaced in future releases. From Cisco IOS XE Release 3.7S and Cisco IOS Release 15.3(1)S both new and legacy commands will coexist until the legacy commands are deprecated in future releases.

Table 1: Replacement Commands Introduced in Cisco IOS XE Release 3.7S and Cisco IOS Release 15.3(1)S

Legacy Command	Replacement Command Introduced in Cisco IOS XE Release 3.7S and Cisco IOS Release 15.3(1)S
backup delay	**redundancy delay (under l2vpn xconnect context)**
bridge-domain (service instance)	**member (bridge-domain)**
clear mpls l2transport fsm state transition	**clear l2vpn atom fsm state transition**
clear mpls l2transport fsm event	**clear l2vpn atom fsm event**
clear xconnect	**clear l2vpn service**
connect (L2VPN local switching)	**l2vpn xconnect context**
debug acircuit	**debug l2vpn acircuit**

Legacy Command	Replacement Command Introduced in Cisco IOS XE Release 3.7S and Cisco IOS Release 15.3(1)S
debug mpls l2transport checkpoint	**debug l2vpn atom checkpoint**
debug mpls l2transport event-trace	**debug l2vpn atom event-trace**
debug mpls l2transport fast-failure-detect	**debug l2vpn atom fast-failure-detect**
debug mpls l2transport signaling	**debug l2vpn atom signaling**
debug mpls l2transport static-oam	**debug l2vpn atom static-oam**
debug mpls l2transport vc subscriber	**debug l2vpn atom vc**
debug mpls l2transport vc	**debug l2vpn atom vc**
debug mpls l2transport vc vccv bfd event	**debug l2vpn atom vc vccv**
debug vfi	**debug l2vpn vfi**
debug vfi checkpoint	**debug l2vpn vfi checkpoint**
debug xconnect	**debug l2vpn xconnect**
debug xconnect rib	**debug l2vpn xconnect rib**
description (L2VFI)	**description (L2VPN)**
l2 pseudowire routing	**pseudowire routing**
l2 router-id	**router-id**
l2 vfi	**l2vpn vfi context**
l2 subscriber	**l2vpn subscriber**
l2 vfi autodiscovery	**autodiscovery**
l2 vfi point-to-point	**l2vpn xconnect context**
local interface	**pseudowire type**
monitor event-trace st-pw-oam	**monitor event-trace pwoam**
mpls label	**label (pseudowire)**
mpls control-word	**control-word (encapsulation mpls under l2vpn connect context)**
neighbor (l2 vfi)	**member (l2vpn vfi)**
protocol	**signaling protocol**
pseudowire-static-oam class	**l2vpn pseudowire static-oam class**

Legacy Command	Replacement Command Introduced in Cisco IOS XE Release 3.7S and Cisco IOS Release 15.3(1)S
pseudowire tlv template	**l2vpn pseudowire tlv template**
pw-class keyword in the **xconnect** command	**source template type pseudowire**
remote link failure notification	**l2vpn remote link failure notification**
show mpls l2transport binding	**show l2vpn atom binding**
show mpls l2transport checkpoint	**show l2vpn atom checkpoint**
show mpls l2transport hw-capability	**show l2vpn atom hw-capability**
show mpls l2transport static-oam	**show l2vpn atom static-oam**
show mpls l2transport summary	**show l2vpn atom summary**
show mpls l2transport pwid	**show l2vpn atom pwid**
show mpls l2transport vc	**show l2vpn atom vc**
show xconnect pwmib	**show l2vpn pwmib**
show xconnect rib	**show l2vpn rib**
show xconnect	**show l2vpn service**
show vfi	**show l2vpn vfi**
xconnect	**l2vpn xconnect context** and **member**
xconnect logging pseudowire status global	**logging pseudowire status**
xconnect logging redundancy global	**logging redundancy**
xconnect *peer-ip vc-id*	**neighbor peer-ip vc-id (xconnect context)**

MPLS L2VPN Protocol-Based CLI: Examples

The examples in this section provide the new configurations that are introduced by the MPLS L2VPN Protocol-Based CLIs feature that replace the existing (legacy) MPLS L2VPN CLIs.

MPLS L2VPN VPWS Configuration Using Replacement (or New) Commands

The following example shows the configuration for Virtual Private Wired Service (VPWS)—Ethernet over Multiprotocol Label Switching (EoMPLS). In this example, L2VPN members point to peer ID or virtual circuit (VC) ID. This configuration is used in most cases except when features like quality of service (QoS), need to be applied at the pseudowire level.

```
l2vpn xconnect context foo
  member GigabitEthernet2/1/1 service-instance 300
  member 10.0.0.1 888 encapsulation mpls
!
interface GigabitEthernet2/1/1
```

```
  service instance 300
   encapsulation dot1q 30
   rewrite ingress tag pop 1 symmetric
  !
  service instance 400
   encapsulation dot1q 40
   rewrite ingress tag pop 1 symmetric

l2vpn xconnect context faa
  member GigabitEthernet2/1/1 service-instance 400
  member 10.0.0.1 999 encapsulation mpls
!
```

MPLS L2VPN Pseudowire Configuration Using Replacement (or New) Commands

In the following example, L2VPN members point to a pseudowire interface. The pseudowire interface is manually configured and includes peer ID and VC ID. This configuration is used in most cases except when features like quality of service (QoS), need to be applied at the pseudowire level.

```
l2vpn xconnect context foo
  member GigabitEthernet2/1/1 service-instance 300
  member Pseudowire888
!
interface Pseudowire 888
 encapsulation mpls
 neighbor 10.0.0.1 888
!
interface Pseudowire 999
 encapsulation mpls
 neighbor 10.0.0.1 999
!
interface GigabitEthernet2/1/1
  service instance 300
   encapsulation dot1q 30
   rewrite ingress tag pop 1 symmetric
  !
  service instance 400
   encapsulation dot1q 40
   rewrite ingress tag pop 1 symmetric

l2vpn xconnect context faa
  member GigabitEthernet2/1/1 service-instance 400
  member Pseudowire 999
!
```

MPLS L2VPN Pseudowire Redundancy Configuration Using Replacement (or New) Commands

The following example shows the configuration for pseudowire redundancy. The new configuration shows concise pseudowire redundancy with no submodes or separate groups. This configuration allows the addition of redundant members to a service without service disruption. This configuration also allows modifying or deleting redundant service configurations without service disruption.

```
l2vpn xconnect context sample-pw-redundancy
  member  service-instance 200
  member 1.1.1.1 180 encap mpls group Denver
  member 2.2.2.2 180180 encap mpls group Denver priority 1
  member 3.3.3.3 180181 encap mpls group Denver priority 2
  redundancy delay 1 20 group Denver
!
interface GigabitEthernet2/1/1
  service instance 200
   encapsulation dot1q 100
   rewrite ingress tag pop 1 symmetric
```

MPLS L2VPN Static Pseudowire Configuration Using Replacement (or New) Commands

Note The following configuration is shown for the Provider Edge (PE) 1 router in a network scheme where Customer Edge (CE) 1 and PE 1 and PE 2 and CE 2 traverse through a Provider core (P) router (CE 1—PE 1—P—PE 2—CE 2).

```
interface g2/1/1
service instance 300 ethernet
encapsulation dot1q 300
no shutdown
!
interface pseudowire 100
neighbor 10.4.4.4 121
encapsulation mpls
label 200 300
signaling protocol none
no shutdown
!
l2vpn xconnect context foo
  member GigabitEthernet2/1/1 service-instance 300
  member pseudowire 100
```

MPLS L2VPN Static Pseudowire Template Configuration Using Replacement (or New) Commands

Note The following configuration is shown for the Provider Edge (PE) 1 router in a network scheme where Customer Edge (CE) 1 and PE 1 and PE 2 and CE 2 traverse through a Provider core (P) router (CE 1—PE 1—P—PE 2—CE 2).

```
template type pseudowire test
encapsulation mpls
signaling protocol none
!
interface g2/1/1
service instance 300 ethernet
encapsulation dot1q 300
no shutdown
!
interface pseudowire 100
neighbor 10.4.4.4 121
source template type pseudowire test
label 200 300
no shutdown
!
l2vpn xconnect context foo
  member GigabitEthernet2/1/1 service-instance 300
  member pseudowire 100
```

MPLS L2VPN Dynamic Pseudowire Template Configuration Using Replacement (or New) Commands

Note The following configuration is shown for the Provider Edge (PE) 1 router in a network scheme where Customer Edge (CE) 1 and PE 1 and PE 2 and CE 2 traverse through a Provider core (P) router (CE 1—PE 1—P—PE 2—CE 2).

```
template type pseudowire test
encapsulation mpls
signaling protocol ldp
!
!
interface g2/1/1
service instance 300 ethernet
encapsulation dot1q 300
no shutdown
!
interface pseudowire 100
neighbor 10.4.4.4 121
source template type pseudowire test
no shutdown
!
l2vpn xconnect context foo
  member GigabitEthernet2/1/1 service-instance 300
  member pseudowire 100
```

MPLS L2VPN Multi-segment Static-Dynamic Pseudowire Template Configuration Using Replacement (or New) Commands

The following PE router configuration is for a multi-segment static-dynamic pseudowire:

```
l2vpn pseudowire tlv template TLV
 tlv mtu 1 4 dec 1500
!
interface pseudowire401
  source template type pseudowire staticTempl
encapsulation mpls
neighbor 10.4.4.4 101
signaling protocol none
label 4401 4301
pseudowire type 4
tlv template TLV
tlv 1 4 dec 1500
tlv vccv-flags C 4 hexstr 0110
!
interface pseudowire501
  source template type pseudowire dynTempl
encapsulation mpls
neighbor 10.2.2.2 101
signaling protocol ldp
```

Displaying MPLS L2VPN Pseudowire Template Configuration Using Replacement (or New) Commands

The following example displays output from the **show interface pseudowire** command:

```
PE1#show interface pseudowire 100
pseudowire100 is up
    Description: Pseudowire Interface
    MTU 1500 bytes, BW 10000000 Kbit
    Encapsulation mpls
    Peer IP 10.4.4.4, VC ID 121
    RX
      21 packets 2623 bytes 0 drops
    TX
      20 packets 2746 bytes 0 drops
```

The following example displays output from the **show template** command:

```
PE1#show template

Template        class/type    Component(s)
ABC             owner        interface pseudowire
 BOUND: pw1
```

Sourcing a Template Under an Interface Pseudowire Using Replacement (or New) Commands

The following example configures the interface pseudowire to inherit all attributes defined from a template on the PE 2 router.

```
PE2(config-subif)#interface pseudowire 100
PE2(config-if)#source template type pseudowire test
PE2(config-if)#neighbor 10.4.4.4 121
PE2(config-if)#no shutdown
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for L2VPN Protocol-Based CLIs

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2: Feature Information for L2VPN Protocol-Based CLIs

Feature Name	Releases	Feature Information
L2VPN Protocol-Based CLIs	Cisco IOS XE Release 3.7S	The L2VPN Protocol-Based CLIs feature provides a set of processes and an improved infrastructure for developing and delivering Cisco IOS software on various Cisco platforms. This feature introduces new commands and modifies or replaces existing commands to achieve a consistent functionality across Cisco platforms and provide cross-Operating System (OS) support. In Cisco IOS XE Release 3.7S, this feature was introduced on the Cisco ASR 903 Router.

CHAPTER 3

Any Transport over MPLS

This module describes how to configure Any Transport over MPLS (AToM) transports data link layer (Layer 2) packets over a Multiprotocol Label Switching (MPLS) backbone. AToM enables service providers to connect customer sites with existing Layer 2 networks by using a single, integrated, packet-based network infrastructure--a Cisco MPLS network. Instead of using separate networks with network management environments, service providers can deliver Layer 2 connections over an MPLS backbone. AToM provides a common framework to encapsulate and transport supported Layer 2 traffic types over an MPLS network core.

AToM supports the following like-to-like transport types:

- ATM Adaptation Layer Type-5 (AAL5) over MPLS
- ATM Cell Relay over MPLS
- Ethernet over MPLS (port modes)

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Any Transport over MPLS

- IP routing must be configured in the core so that the provider edge (PE) routers can reach each other via IP.
- MPLS must be configured in the core so that a label-switched path (LSP) exists between the PE routers.
- A loopback interface must be configured for originating and terminating Layer 2 traffic. Ensure that the PE routers can access the other router's loopback interface. Note that the loopback interface is not needed in all cases. For example, tunnel selection does not need a loopback interface when AToM is directly mapped to a traffic engineering (TE) tunnel.

Restrictions for Any Transport over MPLS

General Restrictions

The following general restrictions pertain to all transport types under AToM:

- Address format: Configure the Label Distribution Protocol (LDP) router ID on all PE routers to be a loopback address with a /32 mask. Otherwise, some configurations might not function properly.

Ethernet over MPLS (EoMPLS) Restrictions

The following restrictions pertain to the Ethernet over MPLS feature:

- Ethernet over MPLS supports VLAN packets that conform to the IEEE 802.1Q standard. The 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames. The Inter-Switch Link (ISL) protocol is not supported between the PE and CE routers.
- The AToM control word is supported. However, if the peer PE does not support a control word, the control word is disabled. This negotiation is done by LDP label binding.
- Ethernet packets with hardware-level cyclic redundancy check (CRC) errors, framing errors, and runt packets are discarded on input.

General Restrictions

- Address format--Configure the Label Distribution Protocol (LDP) router ID on all PE routers to be a loopback address with a /32 mask. Otherwise, some configurations might not function properly.
- For PTPoIP configuration with explicit Null MPLS encapsulation, when a Transparent Clock (TC) is placed between a PTP master and a PTP slave, the TC does not update the correction field.
- If an AToM tunnel spans different service providers that exchange MPLS labels using IPv4 Border Gateway Protocol (BGP) (RFC 3107), you add a label to the stack. The maximum MPLS label stack is .
- Hot standby pseudowire (HSPW) convergence without pseudowire grouping increments linearly. For example, for a thousand virtual circuits, it requires about 54 seconds of convergence time. This is applicable only for the Cisco RSP3 Module.

Clear interface is not the recommended way to measure the convergence numbers.

- With two ECMP paths, load sharing on L2VPN traffic occurs based on odd or even MPLS VC labels. If L2VPN circuits have either odd **or** even MPLS VC labels, load sharing is not performed. But if L2VPN circuits have a combination of both odd **and** even MPLS VC labels, then the odd MPLS VC labels circuits will select one link whereas the even MPLS VC labels circuits will select another link.

ATM AAL5 over MPLS Restrictions

- AAL5 over MPLS is supported only in SDU mode.

ATM Cell Relay over MPLS Restrictions

- If you have TE tunnels running between the PE routers, you must enable LDP on the tunnel interfaces.
- The F4 end-to-end OAM cells are transparently transported along with the ATM cells. When a permanent virtual path (PVP) or permanent virtual circuit (PVC) is down on one PE router, the label associated with that PVP or PVC is withdrawn. Subsequently, the peer PE router detects the label withdrawal and sends an F4 AIS/RDI signal to its corresponding CE router. The PVP or PVC on the peer PE router remains in the up state.
- VC class configuration mode is not supported in port mode.
- The AToM control word is supported. However, if a peer PE does not support the control word, it is disabled.

For configuring ATM cell relay over MPLS in VP mode, the following restrictions apply:

- If a VPI is configured for VP cell relay, you cannot configure a PVC using the same VPI.
- VP trunking (mapping multiple VPs to one emulated VC label) is not supported. Each VP is mapped to one emulated VC.
- VP mode and VC mode drop idle cells.

Ethernet over MPLS (EoMPLS) Restrictions

- The subinterfaces between the CE and PE routers that are running Ethernet over MPLS must be in the same subnet.
- The subinterface on the adjoining CE router must be on the same VLAN as the PE router.
- Ethernet over MPLS supports VLAN packets that conform to the IEEE 802.1Q standard. The 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames. The Inter-Switch Link (ISL) protocol is not supported between the PE and CE routers.
- The AToM control word is supported. However, if the peer PE does not support a control word, the control word is disabled.
- Ethernet packets with hardware-level cyclic redundancy check (CRC) errors, framing errors, and runt packets are discarded on input.

Per-Subinterface MTU for Ethernet over MPLS Restrictions

- The following features do not support MTU values in xconnect subinterface configuration mode:
  - Layer 2 Tunnel Protocol Version 3 (L2TPv3)
  - Virtual Private LAN services (VPLS)
  - L2VPN Pseudowire Switching
- The MTU value can be configured in xconnect subinterface configuration mode only on the following interfaces and subinterfaces:
  - Fast Ethernet
  - Gigabit Ethernet
- The router uses an MTU validation process for remote VCs established through LDP, which compares the MTU value configured in xconnect subinterface configuration mode to the MTU value of the remote customer interface. If an MTU value has not been configured in xconnect subinterface configuration mode, then the validation process compares the MTU value of the local customer interface to the MTU value of the remote xconnect, either explicitly configured or inherited from the underlying interface or subinterface.
- When you configure the MTU value in xconnect subinterface configuration mode, the specified MTU value is not enforced by the dataplane. The dataplane enforces the MTU values of the interface (port mode) or subinterface (VLAN mode).
- Ensure that the interface MTU is larger than the MTU value configured in xconnect subinterface configuration mode. If the MTU value of the customer-facing subinterface is larger than the MTU value of the core-facing interface, traffic may not be able to travel across the pseudowire.

Frame Relay over MPLS Restrictions

Frame Relay traffic shaping is not supported with AToM switched VCs.

HDLC over MPLS Restrictions

- Asynchronous interfaces are not supported.
- You must configure HDLC over MPLS on router interfaces only. You cannot configure HDLC over MPLS on subinterfaces.

PPP over MPLS Restrictions

- Zero hops on one router is not supported. However, you can have back-to-back PE routers.
- Asynchronous interfaces are not supported. The connections between the CE and PE routers on both ends of the backbone must have similar link layer characteristics. The connections between the CE and PE routers must both be synchronous.
- Multilink PPP (MLP) is not supported.
- You must configure PPP on router interfaces only. You cannot configure PPP on subinterfaces.

Tunnel Selection Restrictions

- The selected path should be an LSP destined to the peer PE router.
- The selected tunnel must be an MPLS TE tunnel.
- If you specify an IP address, that address must be the IP address of the loopback interface on the remote PE router. The address must have a /32 mask. There must be an LSP destined to that selected address. The LSP need not be a TE tunnel.

Experimental Bits with AToM Restrictions

- You must statically set the experimental (EXP) bits in both the VC label and the LSP tunnel label, because the LSP tunnel label might be removed at the penultimate router.
- For EXP bits and ATM AAL5 over MPLS and for EXP bits and Frame Relay over MPLS, if you do not assign values to the experimental bits, the priority bits in the header's "tag control information" field are set to zero.
- For EXP bits and ATM Cell Relay over MPLS in VC mode, if you do not assign values to the experimental bits, the priority bits in the header's "tag control information" field are set to zero.
- For EXP bits and HDLC over MPLS and PPP over MPLS, if you do not assign values to the experimental bits, zeros are written into the experimental bit fields.

Remote Ethernet Port Shutdown Restrictions

This feature is not symmetrical if the remote PE router is running an older version image or is on another platform that does not support the EoMPLS remote Ethernet port shutdown feature and the local PE is running an image which supports this feature.

Remote Ethernet Port Shutdown is supported only on EFP with encapsulation default.

Information About Any Transport over MPLS

To configure AToM, you must understand the following concepts:

How AToM Transports Layer 2 Packets

AToM encapsulates Layer 2 frames at the ingress PE and sends them to a corresponding PE at the other end of a pseudowire, which is a connection between the two PE routers. The egress PE removes the encapsulation and sends out the Layer 2 frame.

The successful transmission of the Layer 2 frames between PE routers is due to the configuration of the PE routers. You set up the connection, called a pseudowire, between the routers. You specify the following information on each PE router:

- The type of Layer 2 data that will be transported across the pseudowire, such as Ethernet, Frame Relay, or ATM

- The IP address of the loopback interface of the peer PE router, which enables the PE routers to communicate
- A unique combination of peer PE IP address and VC ID that identifies the pseudowire

The following example shows the basic configuration steps on a PE router that enable the transport of Layer 2 packets. Each transport type has slightly different steps.

Step 1 defines the interface or subinterface on the PE router:

```
Router# interface
 interface-type interface-number
```

Step specifies the encapsulation type for the interface, such as dot1q:

```
Router(config-if-srv)# encapsulation
encapsulation-type
```

Step 4 does the following:

- Makes a connection to the peer PE router by specifying the LDP router ID of the peer PE router.
- Specifies a 32-bit unique identifier, called the VC ID, which is shared between the two PE routers.

The combination of the peer router ID and the VC ID must be unique on the router. Two circuits cannot use the same combination of peer router ID and VC ID.

- Specifies the tunneling method used to encapsulate data in the pseudowire. AToM uses MPLS as the tunneling method.

```
Router(config-if-srv)# xconnect
peer-router-id vcid
encapsulation mpls
```

As an alternative, you can set up a pseudowire class to specify the tunneling method and other characteristics. For more information, see the Configuring the Pseudowire Class, on page 33.

How AToM Transports Layer 2 Packets Using Commands Associated with L2VPN Protocol-Based Feature

AToM encapsulates Layer 2 frames at the ingress PE and sends them to a corresponding PE at the other end of a pseudowire, which is a connection between the two PE routers. The egress PE removes the encapsulation and sends out the Layer 2 frame.

The successful transmission of the Layer 2 frames between PE routers is due to the configuration of the PE routers. You set up the connection, called a pseudowire, between the routers. You specify the following information on each PE router:

- The type of Layer 2 data that will be transported across the pseudowire, such as Ethernet, Frame Relay, or ATM
- The IP address of the loopback interface of the peer PE router, which enables the PE routers to communicate
- A unique combination of peer PE IP address and VC ID that identifies the pseudowire

The following example shows the basic configuration steps on a PE router that enable the transport of Layer 2 packets. Each transport type has slightly different steps.

Step 1 defines the interface or subinterface on the PE router:

```
Router# interface
interface-type interface-number
```

Step 3 specifies the encapsulation type for the interface, such as dot1q:

```
Router(config-if)# encapsulation
encapsulation-type
```

Step 3 does the following:

- Makes a connection to the peer PE router by specifying the LDP router ID of the peer PE router.
- Specifies a 32-bit unique identifier, called the VC ID, which is shared between the two PE routers.

The combination of the peer router ID and the VC ID must be unique on the router. Two circuits cannot use the same combination of peer router ID and VC ID.

- Specifies the tunneling method used to encapsulate data in the pseudowire. AToM uses MPLS as the tunneling method.

```
Router(config)# interface pseudowire 100
Router(config-if)# encapsulation mpls
Router(config-if)# neighbor 10.0.0.1 123
Router(config-if)# exit
!
Router(config)# l2vpn xconnect context A
Router(config-xconnect)# member pseudowire 100


Router(config-xconnect)# exit
```

As an alternative, you can set up a pseudowire class to specify the tunneling method and other characteristics. For more information, see the Configuring the Pseudowire Class, on page 33.

Benefits of AToM

The following list explains some of the benefits of enabling Layer 2 packets to be sent in the MPLS network:

- The AToM product set accommodates many types of Layer 2 packets, including Ethernet and Frame Relay, across multiple Cisco router platforms. This enables the service provider to transport all types of traffic over the backbone and accommodate all types of customers.
- AToM adheres to the standards developed for transporting Layer 2 packets over MPLS. This benefits the service provider that wants to incorporate industry-standard methodologies in the network. Other Layer 2 solutions are proprietary, which can limit the service provider's ability to expand the network and can force the service provider to use only one vendor's equipment.
- Upgrading to AToM is transparent to the customer. Because the service provider network is separate from the customer network, the service provider can upgrade to AToM without disruption of service to the customer. The customers assume that they are using a traditional Layer 2 backbone.

MPLS Traffic Engineering Fast Reroute

AToM can use MPLS traffic engineering (TE) tunnels with fast reroute (FRR) support. AToM VCs can be rerouted around a failed link or node at the same time as MPLS and IP prefixes.

Enabling fast reroute on AToM does not require any special commands; you can use standard fast reroute commands. At the ingress PE, an AToM tunnel is protected by fast reroute when it is routed to an FRR-protected TE tunnel. Both link and node protection are supported for AToM VCs at the ingress PE.

Maximum Transmission Unit Guidelines for Estimating Packet Size

The following calculation helps you determine the size of the packets traveling through the core network. You set the maximum transmission unit (MTU) on the core-facing interfaces of the P and PE routers to accommodate packets of this size. The MTU should be greater than or equal to the total bytes of the items in the following equation:

```
Core MTU >= (Edge MTU + Transport header + AToM header + (MPLS label stack * MPLS label
size))
```

The following sections describe the variables used in the equation.

Edge MTU

The edge MTU is the MTU for the customer-facing interfaces.

Transport Header

The Transport header depends on the transport type. The table below lists the specific sizes of the headers.

Table 3: Header Size of Packets

Transport Type	Packet Size
AAL5	0-32 bytes
Ethernet VLAN	18 bytes
Ethernet Port	14 bytes
Frame Relay DLCI	2 bytes for Cisco encapsulation, 8 bytes for Internet Engineering Task Force (IETF) encapsulation
HDLC	4 bytes
PPP	4 bytes

AToM Header

The AToM header is 4 bytes (control word). The control word is optional for Ethernet, PPP, HDLC, and cell relay transport types. The control word is required for Frame Relay and ATM AAL5 transport types.

MPLS Label Stack

The MPLS label stack size depends on the configuration of the core MPLS network:

- AToM uses one MPLS label to identify the AToM VCs (VC label). Therefore, the minimum MPLS label stack is one for directly connected AToM PEs, which are PE routers that do not have a P router between them.
- If LDP is used in the MPLS network, the label stack size is two (the LDP label and the VC label).
- If a TE tunnel instead of LDP is used between PE routers in the MPLS network, the label stack size is two (the TE label and the VC label).
- If a TE tunnel and LDP are used in the MPLS network (for example, a TE tunnel between P routers or between P and PE routers, with LDP on the tunnel), the label stack is three (TE label, LDP label, VC label).
- If you use MPLS fast reroute in the MPLS network, you add a label to the stack. The maximum MPLS label stack in this case is four (FRR label, TE label, LDP label, VC label).
- If AToM is used by the customer carrier in an MPLS VPN Carrier Supporting Carrier environment, you add a label to the stack. The maximum MPLS label stack in the provider carrier network is .
- If an AToM tunnel spans different service providers that exchange MPLS labels using IPv4 Border Gateway Protocol (BGP) (RFC 3107), you add a label to the stack. The maximum MPLS label stack is
- TE-FRR with BGP labels for layer 2 and layer 3 VPNs must terminate on the BGP gateway because of the four-label limitation.

Other circumstances can increase the MPLS label stack size. Therefore, analyze the complete data path between the AToM tunnel endpoints and determine the maximum MPLS label stack size for your network. Then multiply the label stack size by the size of the MPLS label.

Estimating Packet Size Example

The estimated packet size in the following example is 1526 bytes, based on the following assumptions:

- The edge MTU is 1500 bytes.
- The transport type is Ethernet VLAN, which designates 18 bytes for the transport header.
- The AToM header is 0, because the control word is not used.
- The MPLS label stack is 2, because LDP is used. The MPLS label is 4 bytes.

```
Edge MTU + Transport header + AToM header + (MPLS label stack * MPLS label) = Core MTU
1500     + 18                     + 0      + (2                * 4         ) = 1526
```

You must configure the P and PE routers in the core to accept packets of 1526 bytes.

Per-Subinterface MTU for Ethernet over MPLS

MTU values can be specified in xconnect subinterface configuration mode. When you use xconnect subinterface configuration mode to set the MTU value, you establish a pseudowire connection for situations where the interfaces have different MTU values that cannot be changed.

If you specify an MTU value in xconnect subinterface configuration mode that is outside the range of supported MTU values (64 bytes to the maximum number of bytes supported by the interface), the command might be rejected. If you specify an MTU value that is out of range in xconnect subinterface configuration mode, the router enters the command in subinterface configuration mode.

For example, if you specify an MTU of 1501 in xconnect subinterface configuration mode, and that value is out of range, the router enters the command in subinterface configuration mode, where it is accepted:

```
Router# configure terminal
Router(config)# interface gigabitethernet0/0/2.1
Router(config-subif)# xconnect 10.10.10.1 100 encapsulation mpls
Router(config-subif-xconn)# mtu ?
<64 - 1500> MTU size in bytes
Router(config-subif-xconn)# mtu 1501 <<================
Router(config-subif)# mtu ?
<64 - 17940> MTU size in bytes
```

If the MTU value is not accepted in either xconnect subinterface configuration mode or subinterface configuration mode, then the command is rejected.

Per-Subinterface MTU for Ethernet over MPLS using the commands associated with the L2VPN Protocol-Based CLIs feature

MTU values can be specified in xconnect configuration mode. When you use xconnect configuration mode to set the MTU value, you establish a pseudowire connection for situations where the interfaces have different MTU values that cannot be changed.

If you specify an MTU value in xconnect configuration mode that is outside the range of supported MTU values (64 bytes to the maximum number of bytes supported by the interface), the command might be rejected. If you specify an MTU value that is out of range in xconnect configuration mode, the router enters the command in subinterface configuration mode.

For example, if you specify an MTU of 1501 in xconnect configuration mode, and that value is out of range, the router enters the command in subinterface configuration mode, where it is accepted:

```
Router# configure terminal
Router(config)# interface gigabitethernet0/0/2.1
Router(config)# interface pseudowire 100
Router(config-if)# encapsulation mpls
Router(config-if)# neighbor 10.10.10.1 100
Router(config-if)# mtu ?
<64 - 1500> MTU size in bytes
Router(config-if)# mtu 1501 <<================
Router(config-if)# mtu ?
<64 - 17940> MTU size in bytes
Router(config-if)# exit
!
Router(config)# l2vpn xconnect context A
Router(config-xconnect)# member pseudowire 100 Router
Router(config-xconnect)# member gigabitethernet0/0/2.1
Router(config-xconnect)# exit
```

If the MTU value is not accepted in either xconnect configuration mode or subinterface configuration mode, then the command is rejected.

Frame Relay over MPLS and DTE DCE and NNI Connections

You can configure an interface as a DTE device or a DCE switch, or as a switch connected to a switch with network-to-network interface (NNI) connections. Use the following command in interface configuration mode:

frame-relay intf-type [**dce** | **dte** | **nni**]

The keywords are explained in the table below.

Table 4: frame-relay intf-type Command Keywords

Keyword	Description
dce	Enables the router or access server to function as a switch connected to a router.
dte	Enables the router or access server to function as a DTE device. DTE is the default.
nni	Enables the router or access server to function as a switch connected to a switch.

Local Management Interface and Frame Relay over MPLS

Local Management Interface (LMI) is a protocol that communicates status information about PVCs. When a PVC is added, deleted, or changed, the LMI notifies the endpoint of the status change. LMI also provides a polling mechanism that verifies that a link is up.

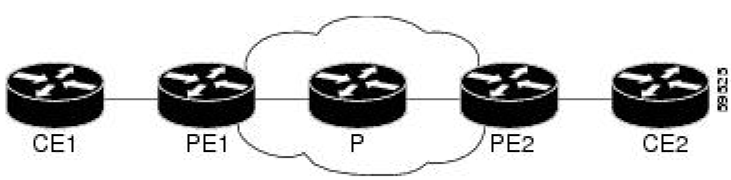
How LMI Works

To determine the PVC status, LMI checks that a PVC is available from the reporting device to the Frame Relay end-user device. If a PVC is available, LMI reports that the status is "Active," which means that all interfaces, line protocols, and core segments are operational between the reporting device and the Frame Relay end-user device. If any of those components is not available, the LMI reports a status of "Inactive."

Note Only the DCE and NNI interface types can report the LMI status.

The figure below is a sample topology that helps illustrate how LMI works.

Figure 1: Sample Topology



In the figure above, note the following:

- CE1 and PE1 and PE2 and CE2 are Frame Relay LMI peers.
- CE1 and CE2 can be Frame Relay switches or end-user devices.
- Each Frame Relay PVC comprises multiple segments.
- The DLCI value is local to each segment and is changed as traffic is switched from segment to segment. Two Frame Relay PVC segments exist in the figure; one is between PE1 and CE1 and the other is between PE2 and CE2.

The LMI protocol behavior depends on whether you have DLCI-to-DLCI or port-to-port connections.

DLCI-to-DLCI Connections

If you have DLCI-to-DLCI connections, LMI runs locally on the Frame Relay ports between the PE and CE devices:

- CE1 sends an active status to PE1 if the PVC for CE1 is available. If CE1 is a switch, LMI checks that the PVC is available from CE1 to the user device attached to CE1.
- PE1 sends an active status to CE1 if the following conditions are met:
  - A PVC for PE1 is available.
  - PE1 received an MPLS label from the remote PE router.
  - An MPLS tunnel label exists between PE1 and the remote PE.

For DTE or DCE configurations, the following LMI behavior exists: The Frame Relay device accessing the network (DTE) does not report the PVC status. Only the network device (DCE) or NNI can report the status. Therefore, if a problem exists on the DTE side, the DCE is not aware of the problem.

Port-to-Port Connections

If you have port-to-port connections, the PE routers do not participate in the LMI status-checking procedures. LMI operates only between the CE routers. The CE routers must be configured as DCE-DTE or NNI-NNI.

For information about LMI, including configuration instructions, see the “Configuring the LMI” section of the Configuring Frame Relay document.

QoS Features Supported with AToM

The tables below list the QoS features supported by AToM.

Table 5: QoS Features Supported with Ethernet over MPLS

QoS Feature	Ethernet over MPLS
Service policy	Can be applied to: • Interface (input and output)
Classification	Supports the following commands: • **match cos** (on interfaces) • **match mpls experimental** (on interfaces) • **match qos-group** (on interfaces) (output policy)
Marking	Supports the following commands: • **set cos** (output policy) • **set discard-class** (input policy) • **set mpls experimental** (input policy) (on interfaces) • **set qos-group** (input policy)

QoS Feature	Ethernet over MPLS
Policing	Supports the following: • Color-aware policing • Multiple-action policing • Single-rate policing • Two-rate policing
Queueing and shaping	Supports the following: • Byte-based WRED • Low Latency Queueing (LLQ) • Weighted Random Early Detection (WRED)

Table 6: QoS Features Supported with Frame Relay over MPLS

QoS Feature	Frame Relay over MPLS
Service policy	Can be applied to: • Interface (input and output) • PVC (input and output)
Classification	Supports the following commands: • **match fr-de** (on interfaces and VCs) • **match fr-dlci** (on interfaces) • **match qos-group**
Marking	Supports the following commands: • **frame-relay congestion management** (output) • **set discard-class** • **set fr-de** (output policy) • **set fr-fecn-becn** (output) • **set mpls experimental** • **set qos-group** • **threshold ecn** (output)

QoS Feature	Frame Relay over MPLS
Policing	Supports the following: • Color-aware policing • Multiple-action policing • Single-rate policing • Two-rate policing
Queueing and shaping	Supports the following: • Byte-based WRED • Class-based weighted fair queueing (CBWFQ) • LLQ • **random-detect discard-class-based** command • Traffic shaping • WRED

Table 7: QoS Features Supported with ATM Cell Relay and AAL5 over MPLS

QoS Feature	ATM Cell Relay and AAL5 over MPLS
Service policy	Can be applied to: • Interface (input and output) • PVC (input and output) • Subinterface (input and output)
Classification	Supports the following commands: • **match mpls experimental** (on VCs) • **match qos-group** (output)
Marking	Supports the following commands: • **random-detect discard-class-based** (input) • **set clp** (output) (on interfaces, subinterfaces, and VCs) • **set discard-class** (input) • **set mpls experimental** (input) (on interfaces, subinterfaces, and VCs) • **set qos-group** (input)

QoS Feature	ATM Cell Relay and AAL5 over MPLS
Policing	Supports the following: • Color-aware policing • Multiple-action policing • Single-rate policing • Two-rate policing
Queueing and shaping	Supports the following: • Byte-based WRED • CBWFQ • Class-based shaping support on ATM PVCs • LLQ • **random-detect discard-class-based** command • WRED

OAM Cell Emulation for ATM AAL5 over MPLS

If a PE router does not support the transport of Operation, Administration, and Maintenance (OAM) cells across a label switched path (LSP), you can use OAM cell emulation to locally terminate or loop back the OAM cells. You configure OAM cell emulation on both PE routers, which emulates a VC by forming two unidirectional LSPs. You use Cisco software commands on both PE routers to enable OAM cell emulation.

After you enable OAM cell emulation on a router, you can configure and manage the ATM VC in the same manner as you would a terminated VC. A VC that has been configured with OAM cell emulation can send loopback cells at configured intervals toward the local CE router. The endpoint can be either of the following:

- End-to-end loopback, which sends OAM cells to the local CE router.
- Segment loopback, which responds to OAM cells to a device along the path between the PE and CE routers.

The OAM cells include the following cells:

- Alarm indication signal (AIS)
- Remote defect indication (RDI)

These cells identify and report defects along a VC. When a physical link or interface failure occurs, intermediate nodes insert OAM AIS cells into all the downstream devices affected by the failure. When a router receives an AIS cell, it marks the ATM VC down and sends an RDI cell to let the remote end know about the failure.

OAM Cell Emulation for ATM AAL5 over MPLS in VC Class Configuration Mode

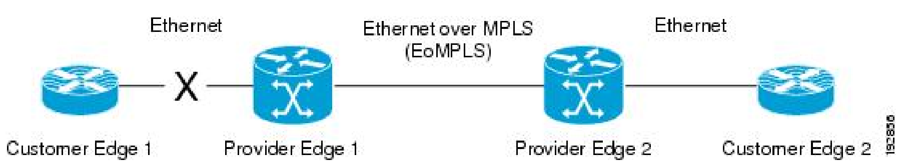
You can configure OAM cell emulation as part of a VC class and then apply the VC class to an interface, a subinterface, or a VC. When you configure OAM cell emulation in VC class configuration mode and then apply the VC class to an interface, the settings in the VC class apply to all the VCs on the interface, unless you specify a different OAM cell emulation value at a lower level, such as the subinterface or VC level. For example, you can create a VC class that specifies OAM cell emulation and sets the rate of AIS cells to every 30 seconds. You can apply the VC class to an interface. Then, for one PVC, you can enable OAM cell emulation and set the rate of AIS cells to every 15 seconds. All the PVCs on the interface use the cell rate of 30 seconds, except for the one PVC that was set to 15 seconds.

Any Transport over MPLS (AToM) Remote Ethernet Port Shutdown

This Cisco IOS XE feature allows a service provider edge (PE) router on the local end of an Ethernet over MPLS (EoMPLS) pseudowire to detect a remote link failure and cause the shutdown of the Ethernet port on the local customer edge (CE) router. Because the Ethernet port on the local CE router is shut down, the router does not lose data by continuously sending traffic to the failed remote link. This is beneficial if the link is configured as a static IP route.

The figure below illustrates a condition in an EoMPLS WAN, with a down Layer 2 tunnel link between a CE router (Customer Edge 1) and the PE router (Provider Edge 1). A CE router on the far side of the Layer 2 tunnel (Customer Edge 2), continues to forward traffic to Customer Edge 1 through the L2 tunnel.

Figure 2: Remote Link Outage in EoMPLS WAN



Previous to this feature, the Provider Edge 2 router could not detect a failed remote link. Traffic forwarded from Customer Edge 2 to Customer Edge 1 would be lost until routing or spanning tree protocols detected the down remote link. If the link was configured with static routing, the remote link outage would be even more difficult to detect.

With this feature, the Provider Edge 2 router detects the remote link failure and causes a shutdown of the local Customer Edge 2 Ethernet port. When the remote L2 tunnel link is restored, the local interface is automatically restored as well. The possibility of data loss is thus diminished.

With reference to the figure above, the Remote Ethernet Shutdown sequence is generally described as follows:

1. The remote link between Customer Edge 1 and Provider Edge 1 fails.
2. Provider Edge 2 detects the remote link failure and disables the transmit laser on the line card interface connected to Customer Edge 2.
3. An RX_LOS error alarm is received by Customer Edge 2 causing Customer Edge 2 to bring down the interface.
4. Provider Edge 2 maintains its interface with Customer Edge 2 in an up state.
5. When the remote link and EoMPLS connection is restored, the Provider Edge 2 router enables the transmit laser.
6. The Customer Edge 2 router brings up its downed interface.

This feature is enabled by default for Ethernet over MPLS (EoMPLS). You can also enable this feature by using the **remote link failure notification** command in xconnect configuration mode as shown in the following example:

```
pseudowire-class eompls
 encapsulation mpls
!
interface GigabitEthernet1/0/0
 xconnect 10.13.13.13 1 pw-class eompls
  remote link failure notification
!
```

This feature can be disabled using the **no remote link failure notification** command in xconnect configuration mode. Use the **show ip interface brief** privileged EXEC command to display the status of all remote L2 tunnel links. Use the **show interface** privileged EXEC command to show the status of the L2 tunnel on a specific interface.

Note The **no remote link failure notification** command will not give notification to clients for remote attachment circuit status down.

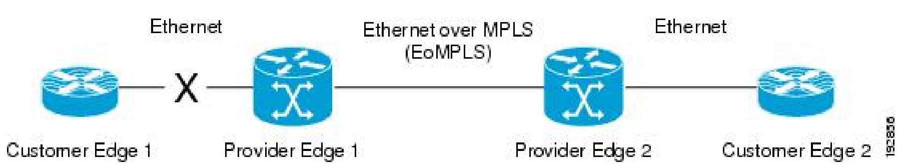
Note Remote Ethernet Port Shutdown is supported only on EFP with encapsulation default.

Any Transport over MPLS (AToM) Remote Ethernet Port Shutdown Using Commands Associated with L2VPN Protocol-Based Feature

This Cisco IOS XE feature allows a service provider edge (PE) router on the local end of an Ethernet over MPLS (EoMPLS) pseudowire to detect a remote link failure and cause the shutdown of the Ethernet port on the local customer edge (CE) router. Because the Ethernet port on the local CE router is shut down, the router does not lose data by continuously sending traffic to the failed remote link. This is beneficial if the link is configured as a static IP route.

The figure below illustrates a condition in an EoMPLS WAN, with a down Layer 2 tunnel link between a CE router (Customer Edge 1) and the PE router (Provider Edge 1). A CE router on the far side of the Layer 2 tunnel (Customer Edge 2), continues to forward traffic to Customer Edge 1 through the L2 tunnel.

Figure 3: Remote Link Outage in EoMPLS WAN



Previous to this feature, the Provider Edge 2 router could not detect a failed remote link. Traffic forwarded from Customer Edge 2 to Customer Edge 1 would be lost until routing or spanning tree protocols detected the down remote link. If the link was configured with static routing, the remote link outage would be even more difficult to detect.

With this feature, the Provider Edge 2 router detects the remote link failure and causes a shutdown of the local Customer Edge 2 Ethernet port. When the remote L2 tunnel link is restored, the local interface is automatically restored as well. The possibility of data loss is thus diminished.

With reference to the figure above, the Remote Ethernet Shutdown sequence is generally described as follows:

1. The remote link between Customer Edge 1 and Provider Edge 1 fails.
2. Provider Edge 2 detects the remote link failure and disables the transmit laser on the line card interface connected to Customer Edge 2.
3. An RX_LOS error alarm is received by Customer Edge 2 causing Customer Edge 2 to bring down the interface.
4. Provider Edge 2 maintains its interface with Customer Edge 2 in an up state.
5. When the remote link and EoMPLS connection is restored, the Provider Edge 2 router enables the transmit laser.
6. The Customer Edge 2 router brings up its downed interface.

This feature is enabled by default for Ethernet over MPLS (EoMPLS). You can also enable this feature by using the **remote link failure notification** command in xconnect configuration mode as shown in the following example:

This feature can be disabled using the **no remote link failure notification** command in xconnect configuration mode. Use the **show ip interface brief** privileged EXEC command to display the status of all remote L2 tunnel links. Use the **show interface** privileged EXEC command to show the status of the L2 tunnel on a specific interface.

Note The **no remote link failure notification** command will not give notification to clients for remote attachment circuit status down.

AToM Load Balancing with Single PW

The AToM Load Balancing with Single PW feature enables load balancing for packets within the same pseudowire by further classifying packets within the same pseudowire into different flows based on certain fields in the packet received on an attachment circuit. For example, for Ethernet this load balancing is based on the source MAC address in the incoming packets.

Flow-Aware Transport (FAT) Load Balancing

The Flow-Aware Transport of MPLS Pseudowires feature enables load balancing of packets within the same pseudowire by further classifying the packets into different flows by adding a flow label at the bottom of the MPLS label stack.

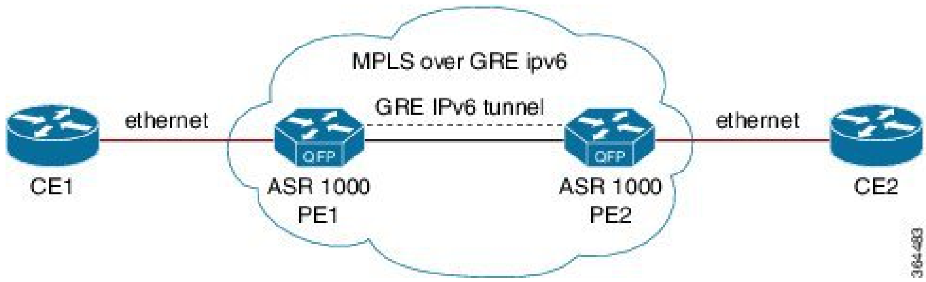
Information About EoMPLS over IPv6 GRE Tunnel

Ethernet over MPLS (EoMPLS) is a tunneling mechanism that allows you to tunnel Layer 2 traffic through a Layer 3 MPLS network. EoMPLS is also known as Layer 2 tunneling.

The EoMPLS over IPv6 GRE Tunnel feature supports tunneling of EoMPLS traffic via an IPv6 network by using GRE tunnels. Effective from Cisco IOS XE Release 3.15s, EoMPLS is supported over IPv6 GRE tunnel.

The following figure shows a deployment model of the EoMPLS over IPv6 GRE Tunnel on a Cisco ASR 1000 Series Aggregation Services Router.

Figure 4: EoMPLS over IPv6 GRE Tunnel Deployment on a Cisco ASR 1000 Series Aggregation Services Router



Additional Information on EoMPLS over IPv6 GRE Tunnel

For more information on EoMPLS over IPv6 GRE Tunnel feature, see GRE IPv6 Tunnels chapter of the *Interface and Hardware Component Configuration Guide, Cisco IOS XE Release 3S (ASR 1000)*.

How to Configure Any Transport over MPLS

This section explains how to perform a basic AToM configuration and includes the following procedures:

Configuring the Pseudowire Class

Note In simple configurations, this task is optional. You need not specify a pseudowire class if you specify the tunneling method as part of the **xconnect** command.

- You must specify the **encapsulation mpls** command as part of the pseudowire class or as part of the **xconnect** command for the AToM VCs to work properly. If you omit the **encapsulation mpls** command as part of the **xconnect** command, you receive the following error:

```
% Incomplete command.
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *name*
4. **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *name* **Example:** `Router(config)# pseudowire-class atom`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.

Configuring the Pseudowire Class Using Commands Associated with L2VPN Protocol-Based Feature

Note In simple configurations, this task is optional. You need not specify a pseudowire class if you specify the tunneling method as part of the **l2vpn xconnect context** command.

- You must specify the **encapsulation mpls** command as part of the pseudowire class or as part of the **l2vpn xconnect context** command for the AToM VCs to work properly. If you omit the **encapsulation mpls** command as part of the **l2vpn xconnect context**command, you receive the following error:

```
% Incomplete command.
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *name*
4. **encapsulation mpls**
5. **neighbor** *peer-address vcid-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface pseudowire** *name* **Example:** `Router(config)# interface pseudowire atom`	Establishes an interface pseudowire with a name that you specify and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 5	**neighbor** *peer-address vcid-value* **Example:** `Router(config-pw-class)# neighbor 33.33.33.33 1`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.

Changing the Encapsulation Type and Removing a Pseudowire

Once you specify the **encapsulation mpls** command, you cannot remove it using the **no encapsulation mpls** command.

Those methods result in the following error message:

```
Encapsulation changes are not allowed on an existing pw-class.
```

To remove the **encapsulation mpls** command, you must delete the pseudowire with the **no pseudowire-class** command.

To change the type of encapsulation, remove the pseudowire using the **no pseudowire-class** command and reconfigure the pseudowire to specify the new encapsulation type.

Changing the Encapsulation Type and Removing a Pseudowire Using Commands Associated with the L2VPN Protocol-Based Feature

Once you specify the **encapsulation mpls** command, you cannot remove it using the **no encapsulation mpls** command.

Those methods result in the following error message:

To remove the **encapsulation mpls** command, you must delete the pseudowire with the **no interface pseudowire** command.

To change the type of encapsulation, remove the pseudowire using the **no template type pseudowire** command and reconfigure the pseudowire to specify the new encapsulation type.

Configuring ATM AAL5 over MPLS

Configuring ATM AAL5 over MPLS on PVCs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
4. **pvc** [*name*] *vpi* / *vci* **l2transport**
5. **encapsulation aal5**
6. **xconnect** *peer-router-id vcid* **encapsulation mpls**
7. **end**
8. **show mpls l2transport vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type and enters interface configuration mode.
Step 4	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 5	**encapsulation aal5** **Example:** `Router(config-if-atm-l2trans-pvc)# encapsulation aal5`	Specifies ATM AAL5 encapsulation for the PVC. Make sure you specify the same encapsulation type on the PE and customer edge (CE) routers.
Step 6	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:**	Binds the attachment circuit to a pseudowire VC.

	Command or Action	Purpose
	`Router(config-if-atm-l2trans-pvc)# xconnect 10.13.13.13 100 encapsulation mpls`	
Step 7	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 8	**show mpls l2transport vc** **Example:** `Router# show mpls l2transport vc`	Displays output that shows ATM AAL5 over MPLS is configured on a PVC.

Examples

The following is sample output from the **show mpls l2transport vc** command that shows that ATM AAL5 over MPLS is configured on a PVC:

```
Router# show mpls l2transport vc
Local intf   Local circuit        Dest address      VC ID      Status
---------    -------------        ------------      -----      ------
ATM1/0       ATM AAL5 1/100       10.4.4.4           100        UP
```

Configuring ATM AAL5 over MPLS on PVCs using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot* / *subslot* / *port*[**.** *subinterface*]
4. **pvc** [*name*] *vpi* / *vci* **l2transport**
5. **encapsulation aal5**
6. **end**
7. **interface pseudowire** *number*
8. **encapsulation mpls**
9. **neighbor** *peer-address vcid-value*
10. **exit**
11. **l2vpn xconnect context** *context-name*
12. **member pseudowire** *interface-number*
13. **member atm** *interface-number* **pvc** *vpi* / *vci*
14. **end**
15. **show l2vpn atom vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot* / *subslot* / *port*[**.** *subinterface*] **Example:** `Device(config)# interface atm1/0/0`	Specifies the interface type and enters interface configuration mode.
Step 4	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Device(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 5	**encapsulation aal5** **Example:** `Device(config-if-atm-l2trans-pvc)# encapsulation aal5`	Specifies ATM AAL5 encapsulation for the PVC. Make sure you specify the same encapsulation type on the PE and customer edge (CE) routers.
Step 6	**end** **Example:** `Device(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 7	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 8	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 9	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.13.13.13 100`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 13	**member atm** *interface-number* **pvc** *vpi* / *vci* **Example:** `Device(config-xconnect)# member atm 100 pvc 1/200`	Specifies the location of the ATM member interface.
Step 14	**end** **Example:** `Device(config-xconnect)# end`	Exits to privileged EXEC mode.
Step 15	**show l2vpn atom vc** **Example:** `Device# show l2vpn atom vc`	Displays output that shows ATM AAL5 over MPLS is configured on a PVC.

Examples

The following is sample output from the **show l2vpn atom vc** command that shows that ATM AAL5 over MPLS is configured on a PVC:

```
Device# show l2vpn atom vc
Local intf   Local circuit          Dest address      VC ID      Status
---------    -------------          ------------      -----      ------
ATM1/0       ATM AAL5 1/100         10.4.4.4           100        UP
```

Configuring ATM AAL5 over MPLS in VC Class Configuration Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **vc-class atm** *vc-class-name*
4. **encapsulation** *layer-type*
5. **exit**
6. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
7. **class-int** *vc-class-name*
8. **pvc** [*name*] *vpi* / *vci* **l2transport**
9. **xconnect** *peer-router-id vcid* **encapsulation mpls**
10. **end**
11. **show atm class-links**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vc-class atm** *vc-class-name* **Example:** `Router(config)# vc-class atm aal5class`	Creates a VC class and enters VC class configuration mode.
Step 4	**encapsulation** *layer-type* **Example:** `Router(config-vc-class)# encapsulation aal5`	Configures the AAL and encapsulation type.
Step 5	**exit** **Example:** `Router(config-vc-class)# exit`	Exits VC class configuration mode.
Step 6	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type enters interface configuration mode.
Step 7	**class-int** *vc-class-name* **Example:** `Router(config-if)# class-int aal5class`	Applies a VC class to the ATM main interface or subinterface. **Note** You can also apply a VC class to a PVC.

	Command or Action	Purpose
Step 8	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 9	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if-atm-l2trans-pvc)# xconnect 10.13.13.13 100 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC.
Step 10	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 11	**show atm class-links** **Example:** `Router# show atm class-links`	Displays the type of encapsulation and that the VC class was applied to an interface.

Examples

In the following example, the command output from the **show atm class-links** command verifies that ATM AAL5 over MPLS is configured as part of a VC class. The command output shows the type of encapsulation and that the VC class was applied to an interface.

```
Router# show atm class-links 1/100
Displaying vc-class inheritance for ATM1/0/0.0, vc 1/100:
no broadcast - Not configured - using default
encapsulation aal5 - VC-class configured on main interface
```

Configuring ATM AAL5 over MPLS in VC Class Configuration Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vc-class atm** *vc-class-name*
4. **encapsulation** *layer-type*
5. **exit**
6. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
7. **class-int** *vc-class-name*
8. **pvc** [*name*] *vpi* / *vci* **l2transport**

9. **exit**
10. **interface pseudowire** *number*
11. **encapsulation mpls**
12. **neighbor** *peer-address vcid-value*
13. **exit**
14. **l2vpn xconnect context** *context-name*
15. **member pseudowire** *interface-number*
16. **member atm** *interface-number*
17. **end**
18. **show atm class-links**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vc-class atm** *vc-class-name* **Example:** `Router(config)# vc-class atm aal5class`	Creates a VC class and enters VC class configuration mode.
Step 4	**encapsulation** *layer-type* **Example:** `Router(config-vc-class)# encapsulation aal5`	Configures the AAL and encapsulation type.
Step 5	**exit** **Example:** `Router(config-vc-class)# exit`	Exits VC class configuration mode.
Step 6	**interface** *type slot* **/** *subslot* **/** *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type enters interface configuration mode.
Step 7	**class-int** *vc-class-name* **Example:**	Applies a VC class to the ATM main interface or subinterface. **Note** You can also apply a VC class to a PVC.

	Command or Action	Purpose
	`Router(config-if)# class-int aal5class`	
Step 8	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 9	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 10	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 11	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 12	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 14	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 15	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 16	**member atm** *interface-number* **Example:** `Device(config-xconnect)# member atm 100`	Specifies the location of the ATM member interface.

	Command or Action	Purpose
Step 17	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 18	**show atm class-links** **Example:** `Router# show atm class-links`	Displays the type of encapsulation and that the VC class was applied to an interface.

Examples

In the following example, the command output from the **show atm class-links** command verifies that ATM AAL5 over MPLS is configured as part of a VC class. The command output shows the type of encapsulation and that the VC class was applied to an interface.

```
Router# show atm class-links 1/100
Displaying vc-class inheritance for ATM1/0/0.0, vc 1/100:
no broadcast - Not configured - using default
encapsulation aal5 - VC-class configured on main interface
```

Configuring OAM Cell Emulation for ATM AAL5 over MPLS

Configuring OAM Cell Emulation for ATM AAL5 over MPLS on PVCs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
4. **pvc** [*name*] *vpi* / *vci* **l2transport**
5. **encapsulation aal5**
6. **xconnect** *peer-router-id vcid* **encapsulation mpls**
7. **oam-ac emulation-enable** [*ais-rate*]
8. *oam-pvc manage* [*frequency*]
9. **end**
10. **show atm pvc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type enters interface configuration mode.
Step 4	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 5	**encapsulation aal5** **Example:** `Router(config-if-atm-l2trans-pvc)# encapsulation aal5`	Specifies ATM AAL5 encapsulation for the PVC. • Specify the same encapsulation type on the PE and CE routers.
Step 6	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if-atm-l2trans-pvc)# xconnect 10.13.13.13 100 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC.
Step 7	**oam-ac emulation-enable** [*ais-rate*] **Example:** `Router(config-if-atm-l2trans-pvc)# oam-ac emulation-enable 30`	Enables OAM cell emulation for AAL5 over MPLS. The *ais-rate* argument lets you specify the rate at which AIS cells are sent. The default is one cell every second. The range is 0 to 60 seconds.
Step 8	*oam-pvc manage* [*frequency*] **Example:** `Router(config-if-atm-l2trans-pvc)# oam-pvc manage`	Enables the PVC to generate end-to-end OAM loopback cells that verify connectivity on the virtual circuit. The optional *frequency* argument is the interval between transmission of loopback cells and ranges from 0 to 600 seconds. The default value is 10 seconds.
Step 9	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.

	Command or Action	Purpose
Step 10	**show atm pvc** **Example:** `Router# show atm pvc`	Displays output that shows OAM cell emulation is enabled on the ATM PVC.

Examples

The following output from the **show atm pvc** command shows that OAM cell emulation is enabled on the ATM PVC:

```
Router# show atm pvc 5/500
ATM4/1/0.200: VCD: 6, VPI: 5, VCI: 500
UBR, PeakRate: 1
AAL5-LLC/SNAP, etype:0x0, Flags: 0x34000C20, VCmode: 0x0
OAM Cell Emulation: enabled, F5 End2end AIS Xmit frequency: 1 second(s)
OAM frequency: 0 second(s), OAM retry frequency: 1 second(s)
OAM up retry count: 3, OAM down retry count: 5
OAM Loopback status: OAM Disabled
OAM VC state: Not ManagedVerified
ILMI VC state: Not Managed
InPkts: 564, OutPkts: 560, InBytes: 19792, OutBytes: 19680
InPRoc: 0, OutPRoc: 0
InFast: 4, OutFast: 0, InAS: 560, OutAS: 560
InPktDrops: 0, OutPktDrops: 0
CrcErrors: 0, SarTimeOuts: 0, OverSizedSDUs: 0
Out CLP=1 Pkts: 0
OAM cells received: 26
F5 InEndloop: 0, F5 InSegloop: 0, F5 InAIS: 0, F5 InRDI: 26
OAM cells sent: 77
F5 OutEndloop: 0, F5 OutSegloop: 0, F5 OutAIS: 77, F5 OutRDI: 0
OAM cell drops: 0
Status: UP
```

Configuring OAM Cell Emulation for ATM AAL5 over MPLS on PVCs using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
4. **pvc** [*name*] *vpi* / *vci* **l2transport**
5. **encapsulation aal5**
6. **exit**
7. **interface pseudowire** *number*
8. **encapsulation mpls**
9. **neighbor** *peer-address vcid-value*
10. **exit**
11. **l2vpn xconnect context** *context-name*

12. **member pseudowire** *interface-number*
13. **member atm** *interface-number* **pvc** *vpi* / *vci*
14. **exit**
15. **pvc** [*name*] *vpi* / *vci* **l2transport**
16. **oam-ac emulation-enable** [*ais-rate*]
17. *oam-pvc manage* [*frequency*]
18. **end**
19. **show atm pvc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type enters interface configuration mode.
Step 4	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 5	**encapsulation aal5** **Example:** `Router(config-if-atm-l2trans-pvc)# encapsulation aal5`	Specifies ATM AAL5 encapsulation for the PVC. • Specify the same encapsulation type on the PE and CE routers.
Step 6	**exit** **Example:** `Router(config-if-atm-l2trans-pvc)# exit`	Exits L2transport PVC configuration mode.
Step 7	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.

	Command or Action	Purpose
Step 8	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 9	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 13	**member atm** *interface-number* **pvc** *vpi* / *vci* **Example:** `Device(config-xconnect)# member atm 100 pvc 1/200`	Specifies the location of the ATM member interface.
Step 14	**exit** **Example:** `Router(config-xconnect)# exit`	Exits xconnect configuration mode.
Step 15	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode.
Step 16	**oam-ac emulation-enable** [*ais-rate*] **Example:** `Router(config-if-atm-l2trans-pvc)# oam-ac emulation-enable 30`	Enables OAM cell emulation for AAL5 over MPLS. The *ais-rate* argument lets you specify the rate at which AIS cells are sent. The default is one cell every second. The range is 0 to 60 seconds.

	Command or Action	Purpose
Step 17	*oam-pvc manage* [*frequency*] **Example:** `Router(config-if-atm-l2trans-pvc)# oam-pvc manage`	Enables the PVC to generate end-to-end OAM loopback cells that verify connectivity on the virtual circuit. The optional *frequency* argument is the interval between transmission of loopback cells and ranges from 0 to 600 seconds. The default value is 10 seconds.
Step 18	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 19	**show atm pvc** **Example:** `Router# show atm pvc`	Displays output that shows OAM cell emulation is enabled on the ATM PVC.

Examples

The following output from the **show atm pvc** command shows that OAM cell emulation is enabled on the ATM PVC:

```
Router# show atm pvc 5/500
ATM4/1/0.200: VCD: 6, VPI: 5, VCI: 500
UBR, PeakRate: 1
AAL5-LLC/SNAP, etype:0x0, Flags: 0x34000C20, VCmode: 0x0
OAM Cell Emulation: enabled, F5 End2end AIS Xmit frequency: 1 second(s)
OAM frequency: 0 second(s), OAM retry frequency: 1 second(s)
OAM up retry count: 3, OAM down retry count: 5
OAM Loopback status: OAM Disabled
OAM VC state: Not ManagedVerified
ILMI VC state: Not Managed
InPkts: 564, OutPkts: 560, InBytes: 19792, OutBytes: 19680
InPRoc: 0, OutPRoc: 0
InFast: 4, OutFast: 0, InAS: 560, OutAS: 560
InPktDrops: 0, OutPktDrops: 0
CrcErrors: 0, SarTimeOuts: 0, OverSizedSDUs: 0
Out CLP=1 Pkts: 0
OAM cells received: 26
F5 InEndloop: 0, F5 InSegloop: 0, F5 InAIS: 0, F5 InRDI: 26
OAM cells sent: 77
F5 OutEndloop: 0, F5 OutSegloop: 0, F5 OutAIS: 77, F5 OutRDI: 0
OAM cell drops: 0
Status: UP
```

Configuring OAM Cell Emulation for ATM AAL5 over MPLS in VC Class Configuration Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **vc-class atm** *name*
4. **encapsulation** *layer-type*
5. **oam-ac emulation-enable** [*ais-rate*]
6. **oam-pvc manage** [*frequency*]
7. **exit**
8. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
9. **class-int** *vc-class-name*
10. **pvc** [*name*] *vpi* / *vci* **l2transport**
11. **xconnect** *peer-router-id vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vc-class atm** *name* **Example:** `Router(config)# vc-class atm oamclass`	Creates a VC class and enters VC class configuration mode.
Step 4	**encapsulation** *layer-type* **Example:** `Router(config-vc-class)# encapsulation aal5`	Configures the AAL and encapsulation type.
Step 5	**oam-ac emulation-enable** [*ais-rate*] **Example:** `Router(config-vc-class)# oam-ac emulation-enable 30`	Enables OAM cell emulation for AAL5 over MPLS and specifies the rate at which AIS cells are sent.
Step 6	**oam-pvc manage** [*frequency*] **Example:** `Router(config-vc-class)# oam-pvc manage`	Enables the PVC to generate end-to-end OAM loopback cells that verify connectivity on the virtual circuit.
Step 7	**exit** **Example:** `Router(config-vc-class)# exit`	Exits VC class configuration mode.

	Command or Action	Purpose
Step 8	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type and enters interface configuration mode.
Step 9	**class-int** *vc-class-name* **Example:** `Router(config-if)# class-int oamclass`	Applies a VC class to the ATM main interface or subinterface. **Note** You can also apply a VC class to a PVC.
Step 10	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 11	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if-atm-l2trans-pvc)# xconnect 10.13.13.13 100 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC.

Configuring OAM Cell Emulation for ATM AAL5 over MPLS in VC Class Configuration Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vc-class atm** *name*
4. **encapsulation** *layer-type*
5. **oam-ac emulation-enable** [*ais-rate*]
6. **oam-pvc manage** [*frequency*]
7. **exit**
8. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
9. **class-int** *vc-class-name*
10. **pvc** [*name*] *vpi* / *vci* **l2transport**
11. **end**
12. **interface pseudowire** *number*
13. **encapsulation mpls**
14. **neighbor** *peer-address vcid-value*
15. **exit**
16. **l2vpn xconnect context** *context-name*
17. **member pseudowire** *interface-number*
18. **member atm** *interface-number*

19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vc-class atm** *name* **Example:** `Router(config)# vc-class atm oamclass`	Creates a VC class and enters VC class configuration mode.
Step 4	**encapsulation** *layer-type* **Example:** `Router(config-vc-class)# encapsulation aal5`	Configures the AAL and encapsulation type.
Step 5	**oam-ac emulation-enable** [*ais-rate*] **Example:** `Router(config-vc-class)# oam-ac emulation-enable 30`	Enables OAM cell emulation for AAL5 over MPLS and specifies the rate at which AIS cells are sent.
Step 6	**oam-pvc manage** [*frequency*] **Example:** `Router(config-vc-class)# oam-pvc manage`	Enables the PVC to generate end-to-end OAM loopback cells that verify connectivity on the virtual circuit.
Step 7	**exit** **Example:** `Router(config-vc-class)# exit`	Exits VC class configuration mode.
Step 8	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type and enters interface configuration mode.
Step 9	**class-int** *vc-class-name* **Example:**	Applies a VC class to the ATM main interface or subinterface. **Note** You can also apply a VC class to a PVC.

	Command or Action	Purpose
	`Router(config-if)# class-int oamclass`	
Step 10	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode. • The **l2transport** keyword indicates that the PVC is a switched PVC instead of a terminated PVC.
Step 11	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 12	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 13	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 14	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 15	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 16	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 17	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 18	**member atm** *interface-number* **Example:** `Device(config-xconnect)# member atm 100`	Specifies the location of the ATM member interface.

	Command or Action	Purpose
Step 19	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring ATM Cell Relay over MPLS

Configuring ATM Cell Relay over MPLS in VC Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot* **/** *subslot* **/** *port* [**.** *subinterface*]
4. **pvc** *vpi* **/** *vci* **l2transport**
5. **encapsulation aal0**
6. **xconnect** *peer-router-id vcid* **encapsulation mpls**
7. **end**
8. **show atm vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot* **/** *subslot* **/** *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies an ATM interface and enters interface configuration mode.
Step 4	**pvc** *vpi* **/** *vci* **l2transport** **Example:** `Router(config-if)# pvc 0/100 l2transport`	Assigns a virtual path identifier (VPI) and virtual circuit identifier (VCI) and enters L2transport VC configuration mode.

	Command or Action	Purpose
Step 5	**encapsulation aal0** **Example:** `Router(config-if-atm-l2trans-pvc)# encapsulation aal0`	For ATM cell relay, specifies raw cell encapsulation for the interface. • Make sure you specify the same encapsulation type on the PE and CE routers.
Step 6	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if-atm-l2trans-pvc)# xconnect 10.13.13.13 100 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC.
Step 7	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 8	**show atm vc** **Example:** `Router# show atm vc`	Verifies that OAM cell emulation is enabled on the ATM VC.

Example

The following sample output from the **show atm vc** command shows that the interface is configured for VC mode cell relay:

```
Router# show atm vc 7
ATM3/0: VCD: 7, VPI: 23, VCI: 100
UBR, PeakRate: 149760
AAL0-Cell Relay, etype:0x10, Flags: 0x10000C2D, VCmode: 0x0
OAM Cell Emulation: not configured
InBytes: 0, OutBytes: 0
Status: UP
```

Configuring ATM Cell Relay over MPLS in VC Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **pvc** *vpi* / *vci* **l2transport**
5. **encapsulation aal0**
6. **end**
7. **interface pseudowire** *number*

8. **encapsulation mpls**
9. **neighbor** *peer-address vcid-value*
10. **exit**
11. **l2vpn xconnect context** *context-name*
12. **member pseudowire** *interface-number*
13. **member atm** *interface-number*
14. **end**
15. **show atm vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies an ATM interface and enters interface configuration mode.
Step 4	**pvc** *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 0/100 l2transport`	Assigns a virtual path identifier (VPI) and virtual circuit identifier (VCI) and enters L2transport VC configuration mode.
Step 5	**encapsulation aal0** **Example:** `Router(config-if-atm-l2trans-pvc)# encapsulation aal0`	For ATM cell relay, specifies raw cell encapsulation for the interface. • Make sure you specify the same encapsulation type on the PE and CE routers.
Step 6	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 7	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.

	Command or Action	Purpose
Step 8	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 9	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 13	**member atm** *interface-number* **Example:** `Device(config-xconnect)# member atm 100`	Specifies the location of the ATM member interface.
Step 14	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.
Step 15	**show atm vc** **Example:** `Router# show atm vc`	Verifies that OAM cell emulation is enabled on the ATM VC.

Example

The following sample output from the **show atm vc** command shows that the interface is configured for VC mode cell relay:

```
Router# show atm vc 7
```

```
ATM3/0: VCD: 7, VPI: 23, VCI: 100
UBR, PeakRate: 149760
AAL0-Cell Relay, etype:0x10, Flags: 0x10000C2D, VCmode: 0x0
OAM Cell Emulation: not configured
InBytes: 0, OutBytes: 0
Status: UP
```

Configuring ATM Cell Relay over MPLS in VC Mode Using VC Class Configuration Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vc-class atm** *name*
4. **encapsulation** *layer-type*
5. **exit**
6. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
7. **class-int** *vc-class-name*
8. **pvc** [*name*] *vpi* / *vci* **l2transport**
9. **xconnect** *peer-router-id vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vc-class atm** *name* **Example:** `Router(config)# vc-class atm cellrelay`	Creates a VC class and enters VC class configuration mode.
Step 4	**encapsulation** *layer-type* **Example:** `Router(config-vc-class)# encapsulation aal0`	Configures the AAL and encapsulation type.
Step 5	**exit** **Example:** `Router(config-vc-class)# exit`	Exits VC class configuration mode.

	Command or Action	Purpose
Step 6	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type and enters interface configuration mode.
Step 7	**class-int** *vc-class-name* **Example:** `Router(config-if)# class-int cellrelay`	Applies a VC class to the ATM main interface or subinterface. **Note** You can also apply a VC class to a PVC.
Step 8	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode.
Step 9	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if-atm-l2trans-pvc)# xconnect 10.13.13.13 100 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC.

Configuring ATM Cell Relay over MPLS in VC Mode Using VC Class Configuration Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vc-class atm** *name*
4. **encapsulation** *layer-type*
5. **exit**
6. **interface** *type slot* / *subslot* / *port* [**.** *subinterface*]
7. **class-int** *vc-class-name*
8. **pvc** [*name*] *vpi* / *vci* **l2transport**
9. **end**
10. **interface pseudowire** *number*
11. **encapsulation mpls**
12. **neighbor** *peer-address vcid-value*
13. **exit**
14. **l2vpn xconnect context** *context-name*
15. **member pseudowire** *interface-number*
16. **member atm** *interface-number*
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vc-class atm** *name* **Example:** `Router(config)# vc-class atm cellrelay`	Creates a VC class and enters VC class configuration mode.
Step 4	**encapsulation** *layer-type* **Example:** `Router(config-vc-class)# encapsulation aal0`	Configures the AAL and encapsulation type.
Step 5	**exit** **Example:** `Router(config-vc-class)# exit`	Exits VC class configuration mode.
Step 6	**interface** *type slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type and enters interface configuration mode.
Step 7	**class-int** *vc-class-name* **Example:** `Router(config-if)# class-int cellrelay`	Applies a VC class to the ATM main interface or subinterface. **Note** You can also apply a VC class to a PVC.
Step 8	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-if)# pvc 1/200 l2transport`	Creates or assigns a name to an ATM PVC and enters L2transport PVC configuration mode.
Step 9	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.

	Command or Action	Purpose
Step 10	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 11	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 12	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 14	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 15	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 16	**member atm** *interface-number* **Example:** `Device(config-xconnect)# member atm 100`	Specifies the location of the ATM member interface.
Step 17	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring ATM Cell Relay over MPLS in PVP Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface atm** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **atm pvp** *vpi* **l2transport**
5. **xconnect** *peer-router-id vcid* **encapsulation mpls**
6. **end**
7. **show atm vp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Defines the interface and enters interface configuration mode.
Step 4	**atm pvp** *vpi* **l2transport** **Example:** `Router(config-if)# atm pvp 1 l2transport`	Specifies that the PVP is dedicated to transporting ATM cells and enters L2transport PVP configuration mode. • The **l2transport** keyword indicates that the PVP is for cell relay. This mode is for Layer 2 transport only; it is not for regular PVPs.
Step 5	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if-atm-l2trans-pvp)# xconnect 10.0.0.1 123 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC. The syntax for this command is the same as for all other Layer 2 transports.
Step 6	**end** **Example:** `Router(config-if-atm-l2trans-pvp)# end`	Exits to privileged EXEC mode.
Step 7	**show atm vp** **Example:** `Router# show atm vp`	Displays output that shows OAM cell emulation is enabled on the ATM VP.

Examples

The following output from the **show atm vp** command shows that the interface is configured for VP mode cell relay:

```
Router# show atm vp 1
ATM5/0  VPI: 1, Cell Relay, PeakRate: 149760, CesRate: 0, DataVCs: 1, CesVCs: 0, Status:
ACTIVE
  VCD    VCI   Type   InPkts   OutPkts   AAL/Encap     Status
  6      3     PVC    0        0         F4 OAM        ACTIVE
  7      4     PVC    0        0         F4 OAM        ACTIVE
TotalInPkts: 0, TotalOutPkts: 0, TotalInFast: 0, TotalOutFast: 0,
TotalBroadcasts: 0 TotalInPktDrops: 0, TotalOutPktDrops: 0
```

Configuring ATM Cell Relay over MPLS in PVP Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **atm pvp** *vpi* **l2transport**
5. **end**
6. **interface pseudowire** *number*
7. **encapsulation mpls**
8. **neighbor** *peer-address vcid-value*
9. **exit**
10. **l2vpn xconnect context** *context-name*
11. **member pseudowire** *interface-number*
12. **member atm** *interface-number* **pvp** *vpi*
13. **end**
14. **show atm vp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface atm** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Defines the interface and enters interface configuration mode.
Step 4	**atm pvp** *vpi* **l2transport** **Example:** `Router(config-if)# atm pvp 1 l2transport`	Specifies that the PVP is dedicated to transporting ATM cells and enters L2transport PVP configuration mode. • The **l2transport** keyword indicates that the PVP is for cell relay. This mode is for Layer 2 transport only; it is not for regular PVPs.
Step 5	**end** **Example:** `Router(config-if-atm-l2trans-pvc)# end`	Exits to privileged EXEC mode.
Step 6	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 8	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 9	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 10	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 11	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.

	Command or Action	Purpose
Step 12	**member atm** *interface-number* **pvp** *vpi* **Example:** `Device(config-xconnect)# member atm 100 pvp 1`	Specifies the location of the ATM member interface.
Step 13	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.
Step 14	**show atm vp** **Example:** `Router# show atm vp`	Displays output that shows OAM cell emulation is enabled on the ATM VP.

Examples

The following output from the **show atm vp** command shows that the interface is configured for VP mode cell relay:

```
Router# show atm vp 1
ATM5/0  VPI: 1, Cell Relay, PeakRate: 149760, CesRate: 0, DataVCs: 1, CesVCs: 0, Status:
ACTIVE
  VCD    VCI   Type   InPkts   OutPkts   AAL/Encap     Status
  6      3     PVC    0        0         F4 OAM        ACTIVE
  7      4     PVC    0        0         F4 OAM        ACTIVE
TotalInPkts: 0, TotalOutPkts: 0, TotalInFast: 0, TotalOutFast: 0,
TotalBroadcasts: 0 TotalInPktDrops: 0, TotalOutPktDrops: 0
```

Configuring Ethernet over MPLS

Configuring Ethernet over MPLS in VLAN Mode to Connect Two VLAN Networks That Are in Different Locations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **encapsulation dot1q** *vlan-id*
5. **xconnect** *peer-router-id vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface gigabitethernet4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. • Make sure the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 4	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets.
Step 5	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-subif)# xconnect 10.0.0.1 123 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC.

Configuring Ethernet over MPLS in VLAN Mode to Connect Two VLAN Networks That Are in Different Locations using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **encapsulation dot1q** *vlan-id*
5. **end**
6. **interface pseudowire** *number*
7. **encapsulation mpls**
8. **neighbor** *peer-address vcid-value*
9. **exit**
10. **l2vpn xconnect context** *context-name*
11. **member pseudowire** *interface-number*
12. **member gigabitethernet** *interface-number*

13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface gigabitethernet4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. • Make sure the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 4	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets.
Step 5	**end** **Example:** `Router(config-subif)# end`	Exits to privileged EXEC mode.
Step 6	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 8	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 9	**exit** **Example:**	Exits interface configuration mode.

	Command or Action	Purpose
	`Router(config-if)# exit`	
Step 10	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 11	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 12	**member gigabitethernet** *interface-number* **Example:** `Router(config-xconnect)# member GigabitEthernet0/0/0.1`	Specifies the location of the Gigabit Ethernet member interface.
Step 13	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring Ethernet over MPLS in Port Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port*
4. **xconnect** *peer-router-id vcid* **encapsulation mpls**
5. **end**
6. **show mpls l2transport vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* **Example:**	Specifies the Gigabit Ethernet interface and enters interface configuration mode.
Step 4	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if)# xconnect 10.0.0.1 123 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 6	**show mpls l2transport vc** **Example:** `Router# show mpls l2transport vc`	Displays information about Ethernet over MPLS port mode.

Configuring Ethernet over MPLS in Port Mode Using Commands Associated with the L2VPN Protocol-Based Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port*[**.** *subinterface*]
4. **end**
5. **interface pseudowire** *number*
6. **encapsulation mpls**
7. **neighbor** *peer-address vcid-value*
8. **exit**
9. **l2vpn xconnect context** *context-name*
10. **member pseudowire** *interface-number*
11. **member gigabitethernet** *interface-number*
12. **end**
13. **end**
14. **show l2vpn atom vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* **/** *subslot* **/** *port*[**.** *subinterface*] **Example:** `Device(config)# interface gigabitethernet4/0/0`	Specifies the Gigabit Ethernet interface and enters interface configuration mode. • Make sure the interface on the adjoining CE router is on the same VLAN as this PE router.
Step 4	**end** **Example:** `Device(config-if)# end`	Exits to privileged EXEC mode.
Step 5	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 6	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 7	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 9	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 10	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.

	Command or Action	Purpose
Step 11	**member gigabitethernet** *interface-number* **Example:** `Device(config-xconnect)# member GigabitEthernet0/0/0.1`	Specifies the location of the Gigabit Ethernet member interface.
Step 12	**end** **Example:** `Device(config-xconnect)# end`	Exits to privileged EXEC mode.
Step 13	**end** **Example:** `Device(config-if)# end`	Exits to privileged EXEC mode.
Step 14	**show l2vpn atom vc** **Example:** `Device# show l2vpn atom vc`	Displays information about Ethernet over MPLS port mode.

Configuring Ethernet over MPLS with VLAN ID Rewrite

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port*
4. **encapsulation dot1q** *vlan-id*
5. **xconnect** *peer-router-id vcid* **encapsulation mpls**
6. **remote circuit id** *remote-vlan-id*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* **Example:**	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode.
Step 4	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets.
Step 5	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-subif)# xconnect 10.0.0.1 123 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC and enters xconnect configuration mode.
Step 6	**remote circuit id** *remote-vlan-id* **Example:** `Router(config-subif-xconn)# remote circuit id 101`	(Optional) Enables you to use VLAN interfaces with different VLAN IDs at both ends of the tunnel.
Step 7	**end** **Example:** `Router(config-subif-xconn)# end`	Exits to privileged EXEC mode.

Configuring Ethernet over MPLS with VLAN ID Rewrite Using Commands Associated with the L2VPN Protocol-Based Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **encapsulation dot1q** *vlan-id*
4. **end**
5. **interface pseudowire** *number*
6. **encapsulation mpls**
7. **neighbor** *peer-address vcid-value*
8. **exit**
9. **l2vpn xconnect context** *context-name*
10. **member pseudowire** *interface-number*
11. **member gigabitethernet** *interface-number*
12. **remote circuit id** *remote-vlan-id*
13. **end**
14. **show controllers eompls forwarding-table**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets.
Step 4	**end** **Example:** `Router(config-subif)# end`	Exits to privileged EXEC mode.
Step 5	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 6	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 7	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 9	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.

	Command or Action	Purpose
Step 10	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 11	**member gigabitethernet** *interface-number* **Example:**	Specifies the location of the Gigabit Ethernet member interface.
Step 12	**remote circuit id** *remote-vlan-id* **Example:** `Router(config-xconnect)# remote circuit id 101`	(Optional) Enables you to use VLAN interfaces with different VLAN IDs at both ends of the tunnel.
Step 13	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.
Step 14	**show controllers eompls forwarding-table** **Example:** `Router# show controllers eompls forwarding-table`	Displays information about VLAN ID rewrite.

Configuring per-Subinterface MTU for Ethernet over MPLS

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **mtu** *mtu-value*
5. **interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*]
6. **encapsulation dot1q** *vlan-id*
7. **xconnect** *peer-router-id vcid* **encapsulation mpls**
8. **mtu** *mtu-value*
9. **end**
10. **show mpls l2transport binding**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface gigabitethernet4/0/0`	Specifies the Gigabit Ethernet interface and enters interface configuration mode.
Step 4	**mtu** *mtu-value* **Example:** `Router(config-if)# mtu 2000`	Specifies the MTU value for the interface. The MTU value specified at the interface level can be inherited by a subinterface.
Step 5	**interface gigabitethernet** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config-if)# interface gigabitethernet4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. Make sure the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 6	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets. The subinterfaces between the CE and PE routers that are running Ethernet over MPLS must be in the same subnet. All other subinterfaces and backbone routers need not be.
Step 7	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-subif)# xconnect 10.0.0.1 123 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC. The syntax for this command is the same as for all other Layer 2 transports. Enters xconnect subinterface configuration mode.
Step 8	**mtu** *mtu-value* **Example:** `Router(config-if-xconn)# mtu 1400`	Specifies the MTU for the VC.
Step 9	**end** **Example:** `Router(config-if-xconn)# end`	Exits to privileged EXEC mode.

	Command or Action	Purpose
Step 10	**show mpls l2transport binding** **Example:** `Router# show mpls l2transport binding`	Displays the MTU values assigned to the local and remote interfaces.

Configuring per-Subinterface MTU for Ethernet over MPLS using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port*[**.** *subinterface*]
4. **mtu** *mtu-value*
5. **interface gigabitethernet** *slot* / *subslot* / *port*[**.** *subinterface*]
6. **encapsulation dot1q** *vlan-id*
7. **end**
8. **interface pseudowire** *number*
9. **encapsulation mpls**
10. **neighbor** *peer-address vcid-value*
11. **mtu** *mtu-value*
12. **exit**
13. **l2vpn xconnect context** *context-name*
14. **member pseudowire** *interface-number*
15. **member gigabitethernet** *interface-number*
16. **end**
17. **show l2vpn atom binding**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port*[**.** *subinterface*] **Example:**	Specifies the Gigabit Ethernet interface and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet4/0/0`	
Step 4	**mtu** *mtu-value* **Example:** `Device(config-if)# mtu 2000`	Specifies the MTU value for the interface. The MTU value specified at the interface level can be inherited by a subinterface.
Step 5	**interface gigabitethernet** *slot* / *subslot* / *port*[**.** *subinterface*] **Example:** `Device(config-if)# interface gigabitethernet4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. Make sure the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 6	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets. The subinterfaces between the CE and PE routers that are running Ethernet over MPLS must be in the same subnet. All other subinterfaces and backbone routers need not be.
Step 7	**end** **Example:** `Device(config-subif)# end`	Exits to privileged EXEC mode.
Step 8	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 9	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 10	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 11	**mtu** *mtu-value* **Example:** `Device(config-if)# mtu 1400`	Specifies the MTU for the VC.
Step 12	**exit** **Example:**	Exits interface configuration mode.

	Command or Action	Purpose
	`Device(config-if)# exit`	
Step 13	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 14	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 15	**member gigabitethernet** *interface-number* **Example:** `Device(config-xconnect)# member GigabitEthernet0/0/0.1`	Specifies the location of the Gigabit Ethernet member interface.
Step 16	**end** **Example:** `Device(config-xconnect)# end`	Exits to privileged EXEC mode.
Step 17	**show l2vpn atom binding** **Example:** `Device# show l2vpn atom binding`	Displays Layer 2 VPN (L2VPN) Any Transport over MPLS (AToM) label binding information.

Configuring Frame Relay over MPLS

Configuring Frame Relay over MPLS with DLCI-to-DLCI Connections

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **frame-relay switching**
4. **interface serial** *slot* / *subslot* / *port* [**.** *subinterface*]
5. **encapsulation frame-relay** [**cisco** | **ietf**]
6. **frame-relay intf-type dce**
7. **exit**
8. **connect** *connection-name interface dlci* **l2transport**
9. **xconnect** *peer-router-id vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**frame-relay switching** **Example:** `Router(config)# frame-relay switching`	Enables PVC switching on a Frame Relay device.
Step 4	**interface serial** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface serial3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 5	**encapsulation frame-relay** [**cisco** | **ietf**] **Example:** `Router(config-if)# encapsulation frame-relay ietf`	Specifies Frame Relay encapsulation for the interface. You can specify different types of encapsulations. You can set one interface to Cisco encapsulation and the other interface to IETF encapsulation.
Step 6	**frame-relay intf-type dce** **Example:** `Router(config-if)# frame-relay intf-type dce`	Specifies that the interface is a DCE switch. You can also specify the interface to support Network-to-Network Interface (NNI) and DTE connections.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits from interface configuration mode.
Step 8	**connect** *connection-name interface dlci* **l2transport** **Example:** `Router(config)# connect fr1 serial5/0 1000 l2transport`	Defines connections between Frame Relay PVCs and enters connect configuration mode. Using the **l2transport** keyword specifies that the PVC will not be a locally switched PVC, but will be tunneled over the backbone network. The *connection-name* argument is a text string that you provide. The *interface* argument is the interface on which a PVC connection will be defined.

	Command or Action	Purpose
		The *dlci* argument is the DLCI number of the PVC that will be connected.
Step 9	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-fr-pw-switching)# xconnect 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets. In a DLCI-to DLCI connection type, Frame Relay over MPLS uses the **xconnect** command in connect configuration mode.

Configuring Frame Relay over MPLS with DLCI-to-DLCI Connections using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **frame-relay switching**
4. **interface serial** *slot* / *subslot* / *port* [**.** *subinterface*]
5. **encapsulation frame-relay** [**cisco** | **ietf**]
6. **frame-relay intf-type dce**
7. **exit**
8. **connect** *connection-name interface dlci* **l2transport**
9. **end**
10. **interface pseudowire** *number*
11. **encapsulation mpls**
12. **neighbor** *peer-address vcid-value*
13. **exit**
14. **l2vpn xconnect context** *context-name*
15. **member pseudowire** *interface-number*
16. **member** *ip-address vc-id* **encapsulation mpls**
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**frame-relay switching** **Example:** `Router(config)# frame-relay switching`	Enables PVC switching on a Frame Relay device.
Step 4	**interface serial** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface serial3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 5	**encapsulation frame-relay** [**cisco** | **ietf**] **Example:** `Router(config-if)# encapsulation frame-relay ietf`	Specifies Frame Relay encapsulation for the interface. You can specify different types of encapsulations. You can set one interface to Cisco encapsulation and the other interface to IETF encapsulation.
Step 6	**frame-relay intf-type dce** **Example:** `Router(config-if)# frame-relay intf-type dce`	Specifies that the interface is a DCE switch. You can also specify the interface to support Network-to-Network Interface (NNI) and DTE connections.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits from interface configuration mode.
Step 8	**connect** *connection-name interface dlci* **l2transport** **Example:** `Router(config)# connect fr1 serial5/0 1000 l2transport`	Defines connections between Frame Relay PVCs and enters connect configuration mode. Using the **l2transport**keyword specifies that the PVC will not be a locally switched PVC, but will be tunneled over the backbone network. The *connection-name*argument is a text string that you provide. The *interface*argument is the interface on which a PVC connection will be defined. The *dlci*argument is the DLCI number of the PVC that will be connected.
Step 9	**end** **Example:** `Router(config-xconnect-conn-config)# end`	Exits to privileged EXEC mode.
Step 10	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.

	Command or Action	Purpose
Step 11	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 12	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 14	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 15	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 16	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 17	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring Frame Relay over MPLS with Port-to-Port Connections

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **encapsulation hdlc**
5. **xconnect** *peer-router-id vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface serial5/0/0`	Specifies a serial interface and enters interface configuration mode.
Step 4	**encapsulation hdlc** **Example:** `Router(config-if)# encapsulation hdlc`	Specifies that Frame Relay PDUs will be encapsulated in HDLC packets.
Step 5	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if)# xconnect 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.

Configuring Frame Relay over MPLS with Port-to-Port Connections using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **encapsulation hdlc**
5. **end**
6. **interface pseudowire** *number*
7. **encapsulation mpls**
8. **neighbor** *peer-address vcid-value*
9. **exit**
10. **l2vpn xconnect context** *context-name*
11. **member pseudowire** *interface-number*
12. **member** *ip-address vc-id* **encapsulation mpls**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface serial5/0/0`	Specifies a serial interface and enters interface configuration mode.
Step 4	**encapsulation hdlc** **Example:** `Router(config-if)# encapsulation hdlc`	Specifies that Frame Relay PDUs will be encapsulated in HDLC packets.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 6	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 8	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 9	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 10	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 11	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 12	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 13	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring HDLC or PPP over MPLS

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot* / *subslot* / *port* [**.** *subinterface*]
4. Do one of the following:
    - **encapsulation ppp**
    - **encapsulation hdlc**
5. **xconnect** *peer-router-id vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**interface serial** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface serial5/0/0`	Specifies a serial interface and enters interface configuration mode.

Step 4	Do one of the following:
	• **encapsulation ppp**
	• **encapsulation hdlc**
	Example:
	`Router(config-if)# encapsulation ppp`
	Example:
	`or`
	Example:
	Example:
	`Router(config-if)# encapsulation hdlc`
		Specifies HDLC or PPP encapsulation and enters connect configuration mode.

Step 5	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-fr-pw-switching)# xconnect 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.

Configuring HDLC or PPP over MPLS using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot* / *subslot* / *port* [**.** *subinterface*]
4. Do one of the following:
    - **encapsulation ppp**
    - **encapsulation hdlc**
5. **end**

6. **interface pseudowire** *number*
7. **encapsulation mpls**
8. **neighbor** *peer-address vcid-value*
9. **exit**
10. **l2vpn xconnect context** *context-name*
11. **member pseudowire** *interface-number*
12. **member** *ip-address vc-id* **encapsulation mpls**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface serial5/0/0`	Specifies a serial interface and enters interface configuration mode.

Step 4	Do one of the following:
	• **encapsulation ppp**
	• **encapsulation hdlc**
	Example:
	`Router(config-if)# encapsulation ppp`
	Example:
	`Router(config-if)# encapsulation hdlc`
		Specifies HDLC or PPP encapsulation and enters connect configuration mode.

Step 5	**end** **Example:** `Router(config-xconnect-conn-config)# end`	Exits to privileged EXEC mode.
Step 6	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.

	Command or Action	Purpose
Step 7	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 8	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 9	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 10	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 11	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 12	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 13	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring Tunnel Selection

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *name*
4. **encapsulation mpls**
5. **preferred-path** {**interface tunnel** *tunnel-number* | **peer**{*ip-address* | *host-name*}} [**disable-fallback**]
6. **exit**

7. **interface** *type slot* / *subslot* / *port*
8. **encapsulation** *encapsulation-type*
9. **xconnect** *peer-router-id vcid* **pw-class** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *name* **Example:** `Router(config)# pseudowire-class ts1`	Establishes a pseudowire class with a name that you specify and enters pseudowire configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation. For AToM, the encapsulation type is mpls.
Step 5	**preferred-path** {**interface tunnel** *tunnel-number* | **peer**{*ip-address* | *host-name*}} [**disable-fallback**] **Example:** `Router(config-pw)# preferred path peer 10.18.18.18`	Specifies the MPLS traffic engineering tunnel or IP address or hostname to be used as the preferred path.
Step 6	**exit** **Example:** `Router(config-pw)# exit`	Exits from pseudowire configuration mode and enables the Tunnel Selection feature.
Step 7	**interface** *type slot* / *subslot* / *port* **Example:** `Router(config)# interface atm1/1/0`	Specifies an interface type and enters interface configuration mode.
Step 8	**encapsulation** *encapsulation-type* **Example:** `Router(config-if)# encapsulation aal5`	Specifies the encapsulation for the interface.
Step 9	**xconnect** *peer-router-id vcid* **pw-class** *name*	Binds the attachment circuit to a pseudowire VC.

	Command or Action	Purpose
	Example: `Router(config-if)# xconnect 10.0.0.1 123 pw-class ts1`	

Examples

In the following sample output from the **show mpls l2transport vc** command includes the following information about the VCs:

- VC 101 has been assigned a preferred path called Tunnel1. The default path is disabled, because the preferred path specified that the default path should not be used if the preferred path fails.
- VC 150 has been assigned an IP address of a loopback address on PE2. The default path can be used if the preferred path fails.

Command output that is in boldface font shows the preferred path information.

```
Router# show mpls l2transport vc detail
Local interface: Gi0/0/0.1 up, line protocol up, Eth VLAN 222 up
  Destination address: 10.16.16.16, VC ID: 101, VC status: up
    Preferred path: Tunnel1,  active
    Default path: disabled
    Tunnel label: 3, next hop point2point
    Output interface: Tu1, imposed label stack {17 16}
  Create time: 00:27:31, last status change time: 00:27:31
  Signaling protocol: LDP, peer 10.16.16.16:0 up
    MPLS VC labels: local 25, remote 16
    Group ID: local 0, remote 6
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 10, send 10
    byte totals:   receive 1260, send 1300
    packet drops:  receive 0, send 0
Local interface: ATM1/0/0 up, line protocol up, ATM AAL5 0/50 up
  Destination address: 10.16.16.16, VC ID: 150, VC status: up
    Preferred path: 10.18.18.18, active
    Default path: ready
    Tunnel label: 3, next hop point2point
    Output interface: Tu2, imposed label stack {18 24}
  Create time: 00:15:08, last status change time: 00:07:37
  Signaling protocol: LDP, peer 10.16.16.16:0 up
    MPLS VC labels: local 26, remote 24
    Group ID: local 2, remote 0
    MTU: local 4470, remote 4470
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 0, send 0
    byte totals:   receive 0, send 0
    packet drops:  receive 0, send 0
```

Troubleshooting Tips

To debug ATM cell packing, issue the **debug atm cell-packing** command.

Configuring Tunnel Selection Using Commands Associated with L2VPN Protocol-Based Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template type pseudowire** *name*
4. **encapsulation mpls**
5. **preferred-path** {**interface tunnel** *tunnel-number* | **peer** {*ip-address* | *hostname*}} [**disable-fallback**]
6. **exit**
7. **interface** *type slot* / *subslot* / *port*[**.** *subinterface*]
8. **encapsulation** *encapsulation-type*
9. **end**
10. **interface pseudowire** *number*
11. **source template type pseudowire** *name*
12. **neighbor** *peer-address vcid-value*
13. **end**
14. **l2vpn xconnect context** *context-name*
15. **member pseudowire** *interface-number*
16. **member** *ip-address vc-id* **encapsulation mpls**
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** *name* **Example:** `Router(config)# template type pseudowire ts1`	Creates a template pseudowire with a name that you specify and enters pseudowire configuration mode.

	Command or Action	Purpose
Step 4	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation. For AToM, the encapsulation type is mpls.
Step 5	**preferred-path** {**interface tunnel** *tunnel-number* | **peer** {*ip-address* | *hostname*}} [**disable-fallback**] **Example:** `Router(config-pw)# preferred path peer 10.18.18.18`	Specifies the MPLS traffic engineering tunnel or IP address or hostname to be used as the preferred path.
Step 6	**exit** **Example:** `Router(config-pw)# exit`	Exits from pseudowire configuration mode and enables the Tunnel Selection feature.
Step 7	**interface** *type slot* **/** *subslot* **/** *port*[**.** *subinterface*] **Example:** `Router(config)# interface atm1/1/0`	Specifies an interface type and enters interface configuration mode.
Step 8	**encapsulation** *encapsulation-type* **Example:** `Router(config-if)# encapsulation aal5`	Specifies the encapsulation for the interface.
Step 9	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 10	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 11	**source template type pseudowire** *name* **Example:** `Router(config-if)# source template type pseudowire ts1`	Configures the source template of type pseudowire named ts1.
Step 12	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)#  neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.

	Command or Action	Purpose
Step 13	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 14	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 15	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 16	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 17	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Troubleshooting Tips using the commands associated with the L2VPN Protocol-Based CLIs feature

You can use the **debug l2vpn atom vc event** command to troubleshoot tunnel selection. For example, if the tunnel interface that is used for the preferred path is shut down, the default path is enabled. The **debug l2vpn atom vc event** command provides the following output:

```
AToM SMGR [10.2.2.2, 101]: Processing imposition update, vc_handle 62091860, update_action
 3, remote_vc_label 16
AToM SMGR [10.2.2.2, 101]: selected route no parent rewrite: tunnel not up
AToM SMGR [10.2.2.2, 101]: Imposition Programmed, Output Interface: Et3/2
```

Setting Experimental Bits with AToM

Note Only EoMPLS and CEM is supported .

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **class-map** *class-name*
4. **match any**
5. **policy-map** *policy-name*
6. **class** *class-name*
7. **set mpls experimental** *value*
8. **exit**
9. **exit**
10. **interface** *type slot* / *subslot* / *port*
11. **service-policy input** *policy-name*
12. **end**
13. **show policy-map interface** *interface-name* [*vc* [*vpi* /] *vci*] [**dlci** *dlci*] [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** *class-name* **Example:** `Router(config)# class-map class1`	Specifies the user-defined name of the traffic class and enters class map configuration mode.
Step 4	**match any** **Example:** `Router(config-cmap)# match any`	Specifies that all packets will be matched. Use only the **any** keyword. Other keywords might cause unexpected results.
Step 5	**policy-map** *policy-name* **Example:** `Router(config-cmap)# policy-map policy1`	Specifies the name of the traffic policy to configure and enters policy-map configuration mode.
Step 6	**class** *class-name* **Example:** `Router(config-pmap)# class class1`	Specifies the name of a predefined traffic class, which was configured with the **class-map** command, used to classify traffic to the traffic policy and enters policy-map class configuration mode.
Step 7	**set mpls experimental** *value* **Example:**	Designates the value to which the MPLS bits are set if the packets match the specified policy map.

	Command or Action	Purpose
	`Router(config-pmap-c)# set mpls experimental 7`	
Step 8	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 9	**exit** **Example:** `Router(config-pmap)# exit`	Exits policy-map configuration mode.
Step 10	**interface** *type* *slot* / *subslot* / *port* **Example:** `Router(config)# interface atm1/0/0`	Specifies the interface type and enters interface configuration mode.
Step 11	**service-policy input** *policy-name* **Example:** `Router(config-if)# service-policy input policy1`	Attaches a traffic policy to an interface.
Step 12	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 13	**show policy-map interface** *interface-name* [*vc* [*vpi* /] *vci*] [**dlci** *dlci*] [**input** | **output**] **Example:** `Router# show policy-map interface serial3/0/0`	Displays the traffic policy attached to an interface.

Enabling the Control Word

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class cw_enable**
4. **encapsulation mpls**
5. **control-word**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class cw_enable** **Example:** `Router(config)# pseudowire-class cw_enable`	Enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is MPLS.
Step 5	**control-word** **Example:** `Router(config-pw-class)# control-word`	Enables the control word.
Step 6	**end** **Example:** `Router(config-pw-class)# end`	Exits to privileged EXEC mode.

Enabling the Control Word using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *number*
4. **encapsulation mpls**
5. **control-word include**
6. **neighbor** *peer-address vcid-value*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 1`	Creates an interface pseudowire with a value that you specify and enters pseudowire configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is mpls.
Step 5	**control-word include** **Example:** `Router(config-pw)# control-word include`	Enables the control word.
Step 6	**neighbor** *peer-address vcid-value* **Example:** `Router(config-pw)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 7	**end** **Example:** `Router(config-pw)# end`	Exits to privileged EXEC mode.

Configuring MPLS AToM Remote Ethernet Port Shutdown

Note The Any Transport over MPLS (AToM): Remote Ethernet Port Shutdown feature is automatically enabled by default when an image with the feature supported is loaded on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation mpls**
5. **exit**
6. **xconnect** *peer-ip-address vc-id pw-class pw-class-name*
7. **no remote link failure notification**
8. **remote link failure notification**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config)# pseudowire-class eompls`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies that MPLS is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.
Step 5	**exit** **Example:** `Router(config-pw)# exit`	Exits to global configuration mode.
Step 6	**xconnect** *peer-ip-address vc-id pw-class pw-class-name* **Example:** `Router(config-if)# xconnect 10.1.1.1 1 pw-class eompls`	Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.
Step 7	**no remote link failure notification** **Example:**	Disables MPLS AToM remote link failure notification and shutdown.

	Command or Action	Purpose
	`Router(config-if-xconn)# remote link failure notification`	
Step 8	**remote link failure notification** **Example:** `Router(config-if-xconn)# remote link failure notification`	Enables MPLS AToM remote link failure notification and shutdown.
Step 9	**end** **Example:** `Router(config-if-xconn)# end`	Exits to privileged EXEC mode.

Configuring MPLS AToM Remote Ethernet Port Shutdown using the commands associated with the L2VPN Protocol-Based CLIs feature

Note The Any Transport over MPLS (AToM): Remote Ethernet Port Shutdown feature is automatically enabled by default when an image with the feature supported is loaded on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template type pseudowire** [*pseudowire-name*]
4. **encapsulation mpls**
5. **exit**
6. **interface** *type slot* / *subslot* / *port*
7. **interface pseudowire** *number*
8. **source template type pseudowire**
9. **neighbor** *peer-address vcid-value*
10. **end**
11. **l2vpn xconnect context** *context-name*
12. **no remote link failure notification**
13. **remote link failure notification**
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** [*pseudowire-name*] **Example:** `Device(config)# template type pseudowire eompls`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw)# encapsulation mpls`	Specifies that MPLS is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.
Step 5	**exit** **Example:** `Device(config-pw)# exit`	Exits to global configuration mode.
Step 6	**interface** *type slot* / *subslot* / *port* **Example:** `Device(config)# interface GigabitEthernet1/0/0`	Configures an interface type and enters interface configuration mode.
Step 7	**interface pseudowire** *number* **Example:** `Device(config-if)# interface pseudowire 100`	Specifies the pseudowire interface.
Step 8	**source template type pseudowire** **Example:** `Device(config-if)# source template type pseudowire eompls`	Configures the source template of type pseudowire named eompls.
Step 9	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.1.1.1 1`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 10	**end** **Example:**	Exits to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**no remote link failure notification** **Example:** `Device(config-xconnect)# no remote link failure notification`	Disables MPLS AToM remote link failure notification and shutdown.
Step 13	**remote link failure notification** **Example:** `Device(config-xconnect)# remote link failure notification`	Enables MPLS AToM remote link failure notification and shutdown.
Step 14	**end** **Example:** `Device(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring AToM Load Balancing with Single PW

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *pw-class-name*
4. **encapsulation mpls**
5. **load-balance flow**
6. **xconnect** *url* **pw-class** *pw-class-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**pseudowire-class** *pw-class-name* **Example:** `Router(config)# pseudowire-class ecmp-class`	Establishes a pseudowire class with a name that you specify, and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is mpls.
Step 5	**load-balance flow** **Example:** `Router(config-pw-class)# load-balance flow`	Enables the AToM Load Balancing with Single PW feature so that load balancing is done on a per-flow basis.
Step 6	**xconnect** *url* **pw-class** *pw-class-name* **Example:** `Router(config-pw-class)# xconnect 10.0.0.1 pw-class ecmp-class`	Binds the attachment circuit to a pseudowire virtual circuit, and enters xconnect configuration mode. • The syntax for this command is the same as for all other Layer 2 transports.

Configuring AToM Load Balancing with Single PW using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template type pseudowire** [*pseudowire-name*]
4. **encapsulation mpls**
5. **load-balance flow**
6. **end**
7. **interface pseudowire** *number*
8. **source template type pseudowire**
9. **neighbor** *peer-address* *vcid-value*
10. **end**
11. **l2vpn xconnect context** *context-name*
12. **member pseudowire** *interface-number*
13. **member** *ip-address vc-id* **encapsulation mpls**
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** [*pseudowire-name*] **Example:** `Router(config)# template type pseudowire eompls`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is mpls.
Step 5	**load-balance flow** **Example:** `Router(config-pw-class)# load-balance flow`	Enables the AToM Load Balancing with Single PW feature so that load balancing is done on a per-flow basis.
Step 6	**end** **Example:** `Router(config-pw-class)# end`	Exits to privileged EXEC mode.
Step 7	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 8	**source template type pseudowire** **Example:** `Router(config-if)# source template type pseudowire ether-pw`	Configures the source template of type pseudowire named ether-pw.
Step 9	**neighbor** *peer-address* *vcid-value* **Example:** `Router(config-if)#  neighbor 10.1.1.1 1`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.

	Command or Action	Purpose
Step 10	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 13	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.1 123 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 14	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring Flow-Aware Transport (FAT) Load Balancing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *name*
4. **encapsulation mpls**
5. **neighbor** *peer-address vcid-value*
6. **signaling protocol ldp**
7. **load-balance flow-label both**
8. **end**
9. **show l2vpn atom vc detail**
10. **show ssm id**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface pseudowire** *name* **Example:** `Device(config)# interface pseudowire 1001`	Establishes a pseudowire with a name that you specify, and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is mpls.
Step 5	**neighbor** *peer-address vcid-value* **Example:** `Device(config-pw-class)# neighbor 10.1.1.200 200`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 6	**signaling protocol ldp** **Example:** `Device(config-pw-class)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.
Step 7	**load-balance flow-label both** **Example:** `Device(config-pw-class)# load-balance flow-label both`	Enables the Flow-Aware Transport of MPLS Pseudowire feature and specifies how flow labels are used. It is recommended that you use both as the option for flow-label. However, if you choose not to use both, you can either use **load-balance flow-label transmit** or **load-balance flow-label receive** if necessary.
Step 8	**end** **Example:** `Device(config-pw-class)# end`	Exits to privileged EXEC mode.
Step 9	**show l2vpn atom vc detail** **Example:** `Device# show l2vpn atom vc detail`	Displays detailed output that shows information about the flow labels configured for the pseudowire.
Step 10	**show ssm id** **Example:**	Displays information for all Segment Switching Manager (SSM) IDs.

	Command or Action	Purpose
	`Device# show ssm id`	

Examples

The following is sample output from the **show mpls l2transport vc 1 detail** command that shows information about the VC details:

```
Device# show mpls l2transport vc 1 detail

Local interface: Te0/5/2 up, line protocol up, Eth VLAN 1 up
  Interworking type is Ethernet
  Destination address: 4.4.4.4, VC ID: 1, VC status: up
    Output interface: BD12, imposed label stack {23 16}
    Preferred path: not configured
    Default path: active
    Next hop: 12.0.0.2
  Create time: 23:12:54, last status change time: 23:09:05
    Last label FSM state change time: 23:09:02
  Signaling protocol: LDP, peer 4.4.4.4:0 up
    Targeted Hello: 1.1.1.1(LDP Id) -> 4.4.4.4, LDP is UP
    Graceful restart: configured and enabled
    Non stop routing: not configured and not enabled
    Status TLV support (local/remote)   : enabled/supported
      LDP route watch                   : enabled
      Label/status state machine        : established, LruRru
      Last local dataplane   status rcvd: No fault
      Last BFD dataplane     status rcvd: Not sent
      Last BFD peer monitor  status rcvd: No fault
      Last local AC  circuit status rcvd: No fault
      Last local AC  circuit status sent: No fault
      Last local PW i/f circ status rcvd: No fault
      Last local LDP TLV     status sent: No fault
      Last remote LDP TLV    status rcvd: No fault
      Last remote LDP ADJ    status rcvd: No fault
    MPLS VC labels: local 27, remote 16
    Group ID: local 8, remote 8
    MTU: local 9216, remote 9216
    Remote interface description:
  Sequencing: receive disabled, send disabled
  Control Word: On
  SSO Descriptor: 4.4.4.4/1, local label: 27
  Dataplane:
    SSM segment/switch IDs: 32854/4116 (used), PWID: 1
  VC statistics:
    transit packet totals: receive 0, send 0
    transit byte totals:   receive 0, send 0
    transit packet drops:  receive 0, seq error 0, send 0
```

The following is sample output from the **show ssm id** command that shows information for all Segment Switching Manager (SSM) IDs:

```
Device# show ssm id

SSM Status: 1 switch
  Switch-ID 4096 State: Open
    Segment-ID: 8194 Type: Eth[2]
```

```
  Switch-ID:                    4096
  Physical intf:                Local
  Allocated By:                 This CPU
  Locked By:                    SIP     [1]
  Circuit status:               UP      [1]
Class:                        SSS
  State:                        Active
  AC Switching Context:         Et0/0
  SSS Info : Switch Handle 2583691265 Ckt 0xC36A59E0
  Interworking 0 Encap Len 0 Boardencap Len 0 MTU 1500
  Flow Classification src-dst-mac
  AC Encap [0 bytes]
Class:                        ADJ
  State:                        Active
  AC Adjacency context:
  adjacency = 0xC36B6100 [complete] RAW Ethernet0/0:0
  AC Encap [0 bytes]
  1stMem: 8194 2ndMem: 0 ActMem: 8194

Segment-ID: 4097 Type: AToM[17]
  Switch-ID:                    4096
  Allocated By:                 This CPU
  Locked By:                    SIP     [1]
Class:                        SSS
  State:                        Active
Class:                        ADJ
  State:                        Active
```

Configuring Flow-Aware Transport (FAT) Load Balancing using a template

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template type pseudowire** [*pseudowire-name*]
4. **encapsulation mpls**
5. **load-balance flow**
6. **load-balance flow-label**
7. **end**
8. **interface pseudowire** *number*
9. **source template type pseudowire**
10. **encapsulation mpls**
11. **neighbor** *peer-address vcid-value*
12. **signaling protocol ldp**
13. **end**
14. **show l2vpn atom vc detail**
15. **show ssm id**
16. **show mpls forwarding-table exact-route**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** [*pseudowire-name*] **Example:** `Device(config)# template type pseudowire fatpw`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is MPLS.
Step 5	**load-balance flow** **Example:** `Device(config-pw-class)# load-balance flow`	Enables the AToM Load Balancing with Single PW feature so that load balancing is done on a per-flow basis.
Step 6	**load-balance flow-label** **Example:** `Device(config-pw-class)# load-balance flow-label both`	Enables the Flow-Aware Transport of MPLS Pseudowires feature and specifies how flow labels are to be used.
Step 7	**end** **Example:** `Device(config-pw-class)# end`	Exits to privileged EXEC mode.
Step 8	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 9	**source template type pseudowire** **Example:** `Device(config-if)# source template type pseudowire fatpw`	Configures the source template of type pseudowire named fatpw.

	Command or Action	Purpose
Step 10	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is MPLS.
Step 11	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.1.1.1 1`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 12	**signaling protocol ldp** **Example:** `Device(config-if)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.
Step 13	**end** **Example:** `Device(config-if)# end`	Exits to privileged EXEC mode.
Step 14	**show l2vpn atom vc detail** **Example:** `Device# show l2vpn atom vc detail`	Displays detailed output that shows information about the flow labels configured for the pseudowire.
Step 15	**show ssm id** **Example:** `Device# show ssm id`	Displays information for all Segment Switching Manager (SSM) IDs.
Step 16	**show mpls forwarding-table exact-route** **Example:** `Device# show mpls forwarding-table exact-route label 32 ethernet source 001d.e558.5c1a dest 000e.8379.1c1b detail`	Displays the exact path for the source and destination address pair.

Examples

The following is sample output from the **show l2vpn atom vc detail** command that shows information about the flow labels configured for the pseudowire:

```
Device# show l2vpn atom vc detail

pseudowire100001 is up, VC status is up PW type: Ethernet
  Create time: 00:01:47, last status change time: 00:01:29
    Last label FSM state change time: 00:01:29
  Destination address: 10.1.1.151 VC ID: 100
```

```
    Output interface: Se3/0, imposed label stack {1001 100}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
    Load Balance: Flow
    flow classification: ethernet src-dst-mac
  Member of xconnect service Et0/0-2, group right
    Associated member Et0/0 is up, status is up
    Interworking type is Like2Like
    Service id: 0xcf000001
  Signaling protocol: LDP, peer 10.1.1.151:0 up
    Targeted Hello: 10.1.1.152(LDP Id) -> 10.1.1.151, LDP is UP
    Graceful restart: not configured and not enabled
    Non stop routing: not configured and not enabled
    PWid FEC (128), VC ID: 100
    Status TLV support (local/remote)         : enabled/supported
      LDP route watch                         : enabled
      Label/status state machine              : established, LruRru
      Local dataplane status received         : No fault
      BFD dataplane status received           : Not sent
      BFD peer monitor status received        : No fault
      Status received from access circuit     : No fault
      Status sent to access circuit           : No fault
      Status received from pseudowire i/f     : No fault
      Status sent to network peer             : No fault
      Status received from network peer       : No fault
      Adjacency status of remote peer         : No fault
  Sequencing: receive disabled, send disabled
  Bindings
    Parameter    Local                          Remote
    ------------ ------------------------------ ------------------------------
    Label        200                            100
    Group ID     0                              0
    Interface
    MTU          1500                           1500
    Control word on (configured: autosense)     on
    PW type      Ethernet                       Ethernet
    VCCV CV type 0x12                           0x12
                   LSPV [2], BFD/Raw [5]          LSPV [2], BFD/Raw [5]
    VCCV CC type 0x07                           0x07
                   CW [1], RA [2], TTL [3]       CW [1], RA [2], TTL [3]
    Status TLV   enabled                        supported
    Flow label   enabled, T=1, R=0              enabled, T=1, R=1
  Dataplane:
    SSM segment/switch IDs: 4097/4096 (used), PWID: 1
  Rx Counters
    28 input transit packets, 2602 bytes
    0 drops, 0 seq err
  Tx Counters
    31 output transit packets, 3694 bytes
    0 drops
```

The following is sample output from the **show ssm id** command that shows information for all Segment Switching Manager (SSM) IDs:

```
Device# show ssm id

SSM Status: 1 switch
  Switch-ID 4096 State: Open
    Segment-ID: 8194 Type: Eth[2]
      Switch-ID:                    4096
      Physical intf:                Local
      Allocated By:                 This CPU
```

```
    Locked By:                     SIP     [1]
    Circuit status:                UP      [1]
  Class:                         SSS
    State:                         Active
    AC Switching Context:          Et0/0
    SSS Info : Switch Handle 2583691265 Ckt 0xC36A59E0
    Interworking 0 Encap Len 0 Boardencap Len 0 MTU 1500
    Flow Classification src-dst-mac
    AC Encap [0 bytes]
  Class:                         ADJ
    State:                         Active
    AC Adjacency context:
    adjacency = 0xC36B6100 [complete] RAW Ethernet0/0:0
    AC Encap [0 bytes]
    1stMem: 8194 2ndMem: 0 ActMem: 8194

  Segment-ID: 4097 Type: AToM[17]
    Switch-ID:                     4096
    Allocated By:                  This CPU
    Locked By:                     SIP     [1]
  Class:                         SSS
    State:                         Active
  Class:                         ADJ
    State:                         Active
```

The following is sample output from the **show mpls forwarding-table exact-route** command that shows the exact path for the source and destination address pair:

```
Device# show mpls forwarding-table exact-route label 32 ethernet source 001d.e558.5c1a dest
 000e.8379.1c1b detail

Local      Outgoing   Prefix           Bytes Label   Outgoing   Next Hop
Label      Label      or Tunnel Id     Switched      interface
32         No Label   l2ckt(66)        1163          Gi1/0/4    point2point
        MAC/Encaps=0/0, MRU=0, Label Stack{}
        No output feature configured
        Flow label: 227190
```

Configuration Examples for Any Transport over MPLS

Example: ATM over MPLS

The table below shows the configuration of ATM over MPLS on two PE routers.

Table 8: ATM over MPLS Configuration Example

PE1	PE2

`mpls label protocol ldp`
` mpls ldp router-id Loopback0 force`
`!`
`interface Loopback0`
` ip address 10.16.12.12 255.255.255.255`
`!`
`interface ATM4/0/0`
` pvc 0/100 l2transport`
`   encapsulation aal0`
`   xconnect 10.13.13.13 100 encapsulation mpls`
`!`
`interface ATM4/0/0.300 point-to-point`
` no ip directed-broadcast`
` no atm enable-ilmi-trap`
` pvc 0/300 l2transport`
`   encapsulation aal0`
`   xconnect 10.13.13.13 300 encapsulation mpls`
	`mpls label protocol ldp`
	` mpls ldp router-id Loopback0 force`
	`!`
	`interface Loopback0`
	` ip address 10.13.13.13 255.255.255.255`
	`interface ATM4/0/0`
	`  pvc 0/100 l2transport`
	`    encapsulation aal0`
	`    xconnect 10.16.12.12 100 encapsulation mpls`
	`!`
	`interface ATM4/0/0.300 point-to-point`
	` no ip directed-broadcast`
	` no atm enable-ilmi-trap`
	` pvc 0/300 l2transport`
	`   encapsulation aal0`
	`   xconnect 10.16.12.12 300 encapsulation mpls`

Example: ATM over MPLS Using Commands Associated with L2VPN Protocol-Based Feature

The table below shows the configuration of ATM over MPLS on two PE routers.

Table 9: ATM over MPLS Configuration Example

PE1	PE2

PE1	PE2
mpls label protocol ldp	mpls label protocol ldp
mpls ldp router-id Loopback0 force	mpls ldp router-id Loopback0 force
!	!
interface Loopback0	interface Loopback0
ip address 10.16.12.12 255.255.255.255	ip address 10.13.13.13 255.255.255.255
!	
interface ATM4/0/0	interface ATM4/0/0
pvc 0/100 l2transport	pvc 0/100 l2transport
encapsulation aal0	encapsulation aal0
interface pseudowire 100	interface pseudowire 100
encapsulation mpls	encapsulation mpls
neighbor 10.0.0.1 123	neighbor 10.0.0.1 123
!	!
l2vpn xconnect context A	l2vpn xconnect context A
member pseudowire 100	member pseudowire 100
member atm 100	member atm 100
!	!
interface ATM4/0/0.300 point-to-point	interface ATM4/0/0.300 point-to-point
no atm enable-ilmi-trap	no ip directed-broadcast
pvc 0/300 l2transport	no atm enable-ilmi-trap
encapsulation aal0	pvc 0/300 l2transport
interface pseudowire 300	encapsulation aal0
encapsulation mpls	interface pseudowire 300
neighbor 10.0.0.1 123	encapsulation mpls

PE1	PE2
!	neighbor 10.0.0.1 123
l2vpn xconnect context A	!
member pseudowire 300	l2vpn xconnect context A
member atm 300	member pseudowire 300
	member atm 300

Example: Configuring ATM AAL5 over MPLS in VC Class Configuration Mode

The following example configures ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to an interface.

```
enable
configure terminal
vc-class atm aal5class
encapsulation aal5
interface atm1/0/0
class-int aal5class
pvc 1/200 l2transport
xconnect 10.13.13.13 100 encapsulation mpls
```

The following example configures ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to a PVC.

```
enable
configure terminal
vc-class atm aal5class
encapsulation aal5
interface atm1/0/0
pvc 1/200 l2transport
class-vc aal5class
xconnect 10.13.13.13 100 encapsulation mpls
```

Example: Configuring ATM AAL5 over MPLS in VC Class Configuration Mode Using Commands Associated with L2VPN Protocol-Based Feature

The following example configures ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to an interface.

```
enable
configure terminal
vc-class atm aal5class
encapsulation aal5
interface atm1/0/0
class-int aal5class
pvc 1/200 l2transport
interface pseudowire 100
encapsulation mpls
```

```
neighbor 10.0.0.1 123
exit
l2vpn xconnect context A
member pseudowire 100
member atm 100
exit
```

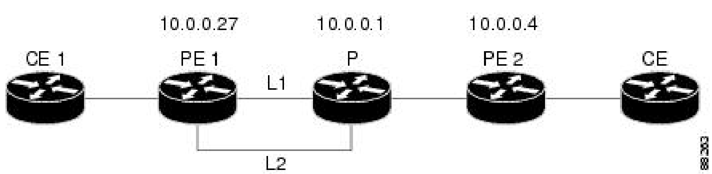
Example: Ethernet over MPLS with MPLS Traffic Engineering Fast Reroute

The following configuration example and the figure show the configuration of Ethernet over MPLS with fast reroute on AToM PE routers.

Routers PE1 and PE2 have the following characteristics:

- A TE tunnel called Tunnel41 is configured between PE1and PE2, using an explicit path through a link called L1. AToM VCs are configured to travel through the FRR-protected tunnel Tunnel41.
- The link L1 is protected by FRR, the backup tunnel is Tunnel1.
- PE2 is configured to forward the AToM traffic back to PE1 through the L2 link.

Figure 5: Fast Reroute Configuration



PE1 Configuration

```
mpls label protocol ldp
mpls traffic-eng tunnels
mpls ldp router-id Loopback1 force
!
pseudowire-class T41
 encapsulation mpls
 preferred-path interface Tunnel41 disable-fallback
!
pseudowire-class IP1
 encapsulation mpls
 preferred-path peer 10.4.0.1 disable-fallback
!
interface Loopback1
 ip address 10.0.0.27 255.255.255.255
!
interface Tunnel1
 ip unnumbered Loopback1
 tunnel destination 10.0.0.1
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 10000
 tunnel mpls traffic-eng path-option 1 explicit name FRR
!
interface Tunnel41
 ip unnumbered Loopback1
 tunnel destination 10.0.0.4
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 1000
```

```
 tunnel mpls traffic-eng path-option 1 explicit name name-1
 tunnel mpls traffic-eng fast-reroute
!
interface POS0/0/0
 description pe1name POS8/0/0
 ip address 10.1.0.2 255.255.255.252
 mpls traffic-eng tunnels
 mpls traffic-eng backup-path Tunnel1
 crc 16
 clock source internal
 pos ais-shut
 pos report lrdi
 ip rsvp bandwidth 155000 155000
!
interface POS0/3/0
 description pe1name POS10/1/0
 ip address 10.1.0.14 255.255.255.252
 mpls traffic-eng tunnels
 crc 16
 clock source internal
 ip rsvp bandwidth 155000 155000
!
interface gigabitethernet3/0/0.1
 encapsulation dot1Q 203
 xconnect 10.0.0.4 2 pw-class IP1
!
interface gigabitethernet3/0/0.2
 encapsulation dot1Q 204
 xconnect 10.0.0.4 4 pw-class T41
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
 mpls traffic-eng router-id Loopback1
 mpls traffic-eng area 0
!
ip classless
ip route 10.4.0.1 255.255.255.255 Tunnel41
!
ip explicit-path name xxxx-1 enable
 next-address 10.4.1.2
 next-address 10.1.0.10
```

P Configuration

```
ip cef
mpls traffic-eng tunnels
!
interface Loopback1
 ip address 10.0.0.1 255.255.255.255
!
interface FastEthernet1/0/0
 ip address 10.4.1.2 255.255.255.0
 mpls traffic-eng tunnels
 ip rsvp bandwidth 10000 10000
!
interface POS8/0/0
 description xxxx POS0/0
 ip address 10.1.0.1 255.255.255.252
 mpls traffic-eng tunnels
 pos ais-shut
 pos report lrdi
 ip rsvp bandwidth 155000 155000
!
```

```
interface POS10/1/0
 description xxxx POS0/3
 ip address 10.1.0.13 255.255.255.252
 mpls traffic-eng tunnels
 ip rsvp bandwidth 155000 155000
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
 mpls traffic-eng router-id Loopback1
 mpls traffic-eng area 0
```

PE2 Configuration

```
ip cef
mpls label protocol ldp
mpls traffic-eng tunnels
mpls ldp router-id Loopback1 force
!
interface Loopback1
 ip address 10.0.0.4 255.255.255.255
!
interface loopback 2
ip address 10.4.0.1 255.255.255.255
!
interface Tunnel27
 ip unnumbered Loopback1
 tunnel destination 10.0.0.27
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 1000
 tunnel mpls traffic-eng path-option 1 explicit name xxxx-1
!
interface FastEthernet0/0/0.2
 encapsulation dot1Q 203
 xconnect 10.0.0.27 2 encapsulation mpls
!
interface FastEthernet0/0/0.3
 encapsulation dot1Q 204
 xconnect 10.0.0.27 4 encapsulation mpls
!
interface FastEthernet1/1/0
 ip address 10.4.1.1 255.255.255.0
 mpls traffic-eng tunnels
 ip rsvp bandwidth 10000 10000
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
 mpls traffic-eng router-id Loopback1
 mpls traffic-eng area 0
!
ip explicit-path name xxxx-1 enable
 next-address 10.4.1.2
 next-address 10.1.0.10
```

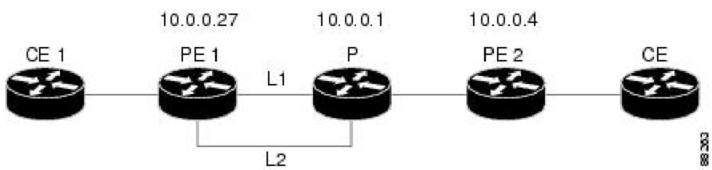
Example: Ethernet over MPLS with MPLS Traffic Engineering Fast Reroute Using Commands Associated with L2VPN Protocol-Based Feature

The following configuration example and the figure show the configuration of Ethernet over MPLS with fast reroute on AToM PE routers.

Routers PE1 and PE2 have the following characteristics:

- A TE tunnel called Tunnel41 is configured between PE1and PE2, using an explicit path through a link called L1. AToM VCs are configured to travel through the FRR-protected tunnel Tunnel41.
- The link L1 is protected by FRR, the backup tunnel is Tunnel1.
- PE2 is configured to forward the AToM traffic back to PE1 through the L2 link.

Figure 6: Fast Reroute Configuration



PE1 Configuration

```
mpls label protocol ldp
mpls traffic-eng tunnels
mpls ldp router-id Loopback1 force
!
template type pseudowire T41
 encapsulation mpls
 preferred-path interface Tunnel41 disable-fallback
!
template type pseudowire IP1
 encapsulation mpls
 preferred-path peer 10.4.0.1 disable-fallback
!
interface Loopback1
 ip address 10.0.0.27 255.255.255.255
!
interface Tunnel1
 ip unnumbered Loopback1
 tunnel destination 10.0.0.1
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 10000
 tunnel mpls traffic-eng path-option 1 explicit name FRR
!
interface Tunnel41
 ip unnumbered Loopback1
 tunnel destination 10.0.0.4
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 1000
 tunnel mpls traffic-eng path-option 1 explicit name name-1
 tunnel mpls traffic-eng fast-reroute
!
interface POS0/0/0
 description pe1name POS8/0/0
 ip address 10.1.0.2 255.255.255.252
 mpls traffic-eng tunnels
 mpls traffic-eng backup-path Tunnel1
 crc 16
 clock source internal
 pos ais-shut
 pos report lrdi
 ip rsvp bandwidth 155000 155000
```

```
!
interface POS0/3/0
 description pe1name POS10/1/0
 ip address 10.1.0.14 255.255.255.252
 mpls traffic-eng tunnels
 crc 16
 clock source internal
 ip rsvp bandwidth 155000 155000
!
interface gigabitethernet3/0/0.1
 encapsulation dot1Q 203
 interface pseudowire 100
 source template type pseudowire T41
 neighbor 10.0.0.4 2
!
l2vpn xconnect context con1
!
interface gigabitethernet3/0/0.2
 encapsulation dot1Q 204
 interface pseudowire 100
 source template type pseudowire IP1
 neighbor 10.0.0.4 4
!
l2vpn xconnect context con2
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
 mpls traffic-eng router-id Loopback1
 mpls traffic-eng area 0
!
ip classless
ip route 10.4.0.1 255.255.255.255 Tunnel41
!
ip explicit-path name xxxx-1 enable
 next-address 10.4.1.2
 next-address 10.1.0.10
```

P Configuration

```
ip cef
mpls traffic-eng tunnels
!
interface Loopback1
 ip address 10.0.0.1 255.255.255.255
!
interface FastEthernet1/0/0
 ip address 10.4.1.2 255.255.255.0
 mpls traffic-eng tunnels
 ip rsvp bandwidth 10000 10000
!
interface POS8/0/0
 description xxxx POS0/0
 ip address 10.1.0.1 255.255.255.252
 mpls traffic-eng tunnels
 pos ais-shut
 pos report lrdi
 ip rsvp bandwidth 155000 155000
!
interface POS10/1/0
 description xxxx POS0/3
 ip address 10.1.0.13 255.255.255.252
 mpls traffic-eng tunnels
 ip rsvp bandwidth 155000 155000
```

```
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
 mpls traffic-eng router-id Loopback1
 mpls traffic-eng area 0
```

PE2 Configuration

```
ip cef
mpls label protocol ldp
mpls traffic-eng tunnels
mpls ldp router-id Loopback1 force
!
interface Loopback1
 ip address 10.0.0.4 255.255.255.255
!
interface loopback 2
ip address 10.4.0.1 255.255.255.255
!
interface Tunnel27
 ip unnumbered Loopback1
 tunnel destination 10.0.0.27
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 1000
 tunnel mpls traffic-eng path-option 1 explicit name xxxx-1
!
interface FastEthernet0/0/0.2
 encapsulation dot1Q 203
 interface pseudowire 100
 encapsulation mpls
 neighbor 10.0.0.1 123
!
l2vpn xconnect context A
 member pseudowire 100
 member gigabitethernet 0/0/0.1
!
interface FastEthernet0/0/0.3
 encapsulation dot1Q 204
 interface pseudowire 100
 encapsulation mpls
 neighbor 10.0.0.1 123
!
l2vpn xconnect context A
 member pseudowire 100
 member gigabitethernet 0/0/0.1
!
interface FastEthernet1/1/0
 ip address 10.4.1.1 255.255.255.0
 mpls traffic-eng tunnels
 ip rsvp bandwidth 10000 10000
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
 mpls traffic-eng router-id Loopback1
 mpls traffic-eng area 0
!
ip explicit-path name xxxx-1 enable
 next-address 10.4.1.2
 next-address 10.1.0.10
```

Example: Configuring OAM Cell Emulation

The following example shows how to enable OAM cell emulation on an ATM PVC:

```
interface ATM 1/0/0
pvc 1/200 l2transport
encapsulation aal5
xconnect 10.13.13.13 100 encapsulation mpls
oam-ac emulation-enable
oam-pvc manage
```

The following example shows how to set the rate at which an AIS cell is sent every 30 seconds:

```
interface ATM 1/0/0
pvc 1/200 l2transport
encapsulation aal5
xconnect 10.13.13.13 100 encapsulation mpls
oam-ac emulation-enable 30
oam-pvc manage
```

The following example shows how to configure OAM cell emulation for ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to an interface.

```
enable
configure terminal
vc-class atm oamclass
encapsulation aal5
oam-ac emulation-enable 30
oam-pvc manage
interface atm1/0/0
class-int oamclass
pvc 1/200 l2transport
xconnect 10.13.13.13 100 encapsulation mpls
```

The following example shows how to configure OAM cell emulation for ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to a PVC.

```
enable
configure terminal
vc-class atm oamclass
encapsulation aal5
oam-ac emulation-enable 30
oam-pvc manage
interface atm1/0/0
pvc 1/200 l2transport
class-vc oamclass
xconnect 10.13.13.13 100 encapsulation mpls
```

The following example shows how to configure OAM cell emulation for ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to an interface. One PVC is configured with OAM cell emulation at an AIS rate of 10. That PVC uses the AIS rate of 10 instead of 30.

```
enable
configure terminal
vc-class atm oamclass
encapsulation aal5
oam-ac emulation-enable 30
oam-pvc manage
```

```
interface atm1/0/0
class-int oamclass
pvc 1/200 l2transport
oam-ac emulation-enable 10
xconnect 10.13.13.13 100 encapsulation mpls
```

Example: Configuring OAM Cell Emulation using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows how to enable OAM cell emulation on an ATM PVC:

```
interface ATM 1/0/0
pvc 1/200 l2transport
encapsulation aal5
interface pseudowire 100
encapsulation mpls
neighbor 10.0.0.1 123
!
l2vpn xconnect context A
member pseudowire 100
member gigabitethernet 0/0/0.1
!
oam-ac emulation-enable
oam-pvc manage
```

The following example shows how to set the rate at which an AIS cell is sent every 30 seconds:

```
interface ATM 1/0/0
pvc 1/200 l2transport
encapsulation aal5
interface pseudowire 100
encapsulation mpls
neighbor 10.0.0.1 123
!
l2vpn xconnect context A
member pseudowire 100
member gigabitethernet 0/0/0.1
!
oam-ac emulation-enable 30
oam-pvc manage
```

The following example shows how to configure OAM cell emulation for ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to an interface.

```
enable
configure terminal
vc-class atm oamclass
encapsulation aal5
oam-ac emulation-enable 30
oam-pvc manage
interface atm1/0/0
class-int oamclass
pvc 1/200 l2transport
interface pseudowire 100
encapsulation mpls
neighbor 10.0.0.1 123
!
l2vpn xconnect context A
```

```
member pseudowire 100
member gigabitethernet 0/0/0.1

The following example shows how to configure OAM cell emulation for ATM AAL5 over MPLS in
VC class configuration mode. The VC class is then applied to a PVC.
```

```
enable
configure terminal
vc-class atm oamclass
encapsulation aal5
oam-ac emulation-enable 30
oam-pvc manage
interface atm1/0/0
pvc 1/200 l2transport
class-vc oamclass
interface pseudowire 100
encapsulation mpls
neighbor 10.0.0.1 123
!
l2vpn xconnect context A
member pseudowire 100
member gigabitethernet 0/0/0.1
```

The following example shows how to configure OAM cell emulation for ATM AAL5 over MPLS in VC class configuration mode. The VC class is then applied to an interface. One PVC is configured with OAM cell emulation at an AIS rate of 10. That PVC uses the AIS rate of 10 instead of 30.

```
enable
configure terminal
vc-class atm oamclass
encapsulation aal5
oam-ac emulation-enable 30
oam-pvc manage
interface atm1/0/0
class-int oamclass
pvc 1/200 l2transport
oam-ac emulation-enable 10
interface pseudowire 100
encapsulation mpls
neighbor 10.0.0.1 123
!
l2vpn xconnect context A
member pseudowire 100
member gigabitethernet 0/0/0.1
```

Example: Configuring ATM Cell Relay over MPLS

The following example shows how to configure ATM cell relay over MPLS in VC class configuration mode. The VC class is then applied to an interface.

```
enable
configure terminal
vc-class atm cellrelay
encapsulation aal0
interface atm1/0/0
class-int cellrelay
pvc 1/200 l2transport
xconnect 10.13.13.13 100 encapsulation mpls
```

The following example shows how to configure ATM cell relay over MPLS in VC class configuration mode. The VC class is then applied to a PVC.

```
enable
configure terminal
vc-class atm cellrelay
encapsulation aal0
interface atm1/0/0
pvc 1/200 l2transport
class-vc cellrelay
xconnect 10.13.13.13 100 encapsulation mpls
```

The following example shows how to configure a pseudowire class to transport single ATM cells over a virtual path:

```
pseudowire-class vp-cell-relay
encapsulation mpls
interface atm 5/0
atm pvp 1 l2transport
xconnect 10.0.0.1 123 pw-class vp-cell-relay
```

Example: Configuring ATM Cell Relay over MPLS using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows how to configure ATM cell relay over MPLS in VC class configuration mode. The VC class is then applied to an interface.

```
enable
configure terminal
vc-class atm cellrelay
encapsulation aal0
interface atm1/0/0
class-int cellrelay
pvc 1/200 l2transport
interface pseudowire 100
encapsulation mpls
neighbor 10.13.13.13 100
!
l2vpn xconnect context A
member pseudowire 100
member gigabitethernet 0/0/0.1
```

The following example shows how to configure ATM cell relay over MPLS in VC class configuration mode. The VC class is then applied to a PVC.

```
enable
configure terminal
vc-class atm cellrelay
encapsulation aal0
interface atm1/0/0
pvc 1/200 l2transport
class-vc cellrelay
interface pseudowire 100
encapsulation mpls
neighbor 10.13.13.13 100
!
l2vpn xconnect context A
```

```
member pseudowire 100
member gigabitethernet 0/0/0.1
```

The following example shows how to configure a pseudowire class to transport single ATM cells over a virtual path:

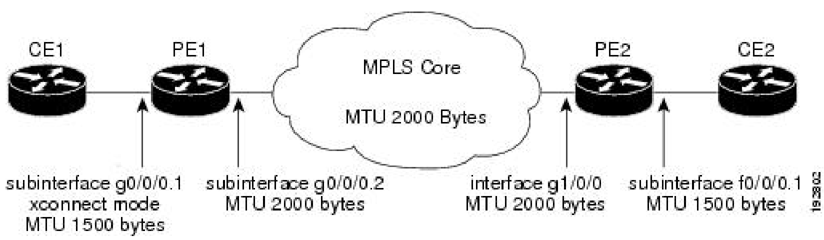
```
template type pseudowire vp-cell-relay
encapsulation mpls
interface atm 5/0
atm pvp 1 l2transport
interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.0.0.1 123
!
l2vpn xconnect context con1
```

Example: Configuring per-Subinterface MTU for Ethernet over MPLS

The figure below shows a configuration that enables matching MTU values between VC endpoints.

As shown in the figure, PE1 is configured in xconnect subinterface configuration mode with an MTU value of 1500 bytes in order to establish an end-to-end VC with PE2, which also has an MTU value of 1500 bytes. If PE1 was not set with an MTU value of 1500 bytes, in xconnect subinterface configuration mode, the subinterface would inherit the MTU value of 2000 bytes set on the interface. This would cause a mismatch in MTU values between the VC endpoints, and the VC would not come up.

Figure 7: Configuring MTU Values in xconnect Subinterface Configuration Mode



The following examples show the router configurations in the figure above:

CE1 Configuration

```
interface gigabitethernet0/0/0
 mtu 1500
 no ip address
!
interface gigabitethernet0/0/0.1
 encapsulation dot1Q 100
 ip address 10.181.182.1 255.255.255.0
```

PE1 Configuration

```
interface gigabitethernet0/0/0
 mtu 2000
 no ip address
!
interface gigabitethernet0/0/0.1
 encapsulation dot1Q 100
```

```
 xconnect 10.1.1.152 100 encapsulation mpls
  mtu 1500
!
interface gigabitethernet0/0/0.2
 encapsulation dot1Q 200
 ip address 10.151.100.1 255.255.255.0
 mpls ip
```

PE2 Configuration

```
interface gigabitethernet1/0/0
 mtu 2000
 no ip address
!
interface gigabitethernet1/0/0.2
 encapsulation dot1Q 200
 ip address 10.100.152.2 255.255.255.0
 mpls ip
!
interface fastethernet0/0/0
 no ip address
!
interface fastethernet0/0/0.1
 description default MTU of 1500 for FastEthernet
 encapsulation dot1Q 100
 xconnect 10.1.1.151 100 encapsulation mpls
```

CE2 Configuration

```
interface fastethernet0/0/0
 no ip address
interface fastethernet0/0/0.1
 encapsulation dot1Q 100
 ip address 10.181.182.2 255.255.255.0
```

The **show mpls l2transport binding** command, issued from router PE1, shows a matching MTU value of 1500 bytes on both the local and remote routers:

```
Router# show mpls l2transport binding
Destination Address: 10.1.1.152,  VC ID: 100
    Local Label: 100
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]
    Remote Label: 202
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: RA [2]
              CV Type: LSPV [2]


Router# show mpls l2transport vc detail
Local interface: Gi0/0/0.1 up, line protocol up, Eth VLAN 100 up
  Destination address: 10.1.1.152, VC ID: 100, VC status: up
    Output interface: Gi0/0/0.2, imposed label stack {202}
    Preferred path: not configured
    Default path: active
    Next hop: 10.151.152.2
  Create time: 1d11h, last status change time: 1d11h
  Signaling protocol: LDP, peer 10.1.1.152:0 up
```

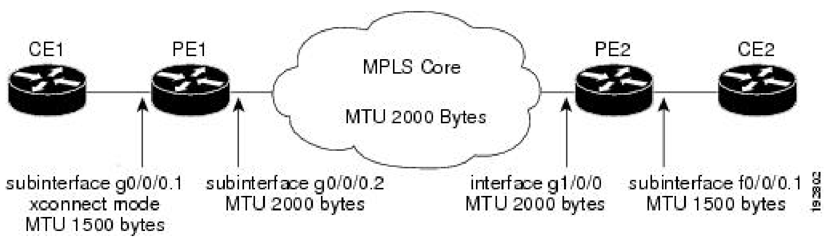
```
    Targeted Hello: 10.1.1.151(LDP Id) -> 10.1.1.152
    MPLS VC labels: local 100, remote 202
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 41, send 39
    byte totals:   receive 4460, send 5346
    packet drops:  receive 0, send 0
```

Example: Configuring per-Subinterface MTU for Ethernet over MPLS using the commands associated with the L2VPN Protocol-Based CLIs feature

The figure below shows a configuration that enables matching MTU values between VC endpoints.

As shown in the figure, PE1 is configured in xconnect subinterface configuration mode with an MTU value of 1500 bytes in order to establish an end-to-end VC with PE2, which also has an MTU value of 1500 bytes. If PE1 was not set with an MTU value of 1500 bytes, in xconnect subinterface configuration mode, the subinterface would inherit the MTU value of 2000 bytes set on the interface. This would cause a mismatch in MTU values between the VC endpoints, and the VC would not come up.

Figure 8: Configuring MTU Values in xconnect Subinterface Configuration Mode



The following examples show the router configurations in the figure above:

CE1 Configuration

```
interface gigabitethernet0/0/0
 mtu 1500
 no ip address
!
interface gigabitethernet0/0/0.1
 encapsulation dot1Q 100
 ip address 10.181.182.1 255.255.255.0
```

PE1 Configuration

```
interface gigabitethernet0/0/0
 mtu 2000
 no ip address
!
interface gigabitethernet0/0/0.1
 encapsulation dot1Q 100
 interface pseudowire 100
encapsulation mpls
neighbor 10.0.0.1 123
mtu 1500
```

```
!
l2vpn xconnect context A
member pseudowire 100
member gigabitethernet 0/0/0.1
!
interface gigabitethernet0/0/0.2
 encapsulation dot1Q 200
 ip address 10.151.100.1 255.255.255.0
 mpls ip
```

PE2 Configuration

```
interface gigabitethernet1/0/0
 mtu 2000
 no ip address
!
interface gigabitethernet1/0/0.2
 encapsulation dot1Q 200
 ip address 10.100.152.2 255.255.255.0
 mpls ip
!
interface fastethernet0/0/0
 no ip address
!
interface fastethernet0/0/0.1
 description default MTU of 1500 for FastEthernet
 encapsulation dot1Q 100
 interface pseudowire 100
 encapsulation mpls
 neighbor 10.0.0.1 123
mtu 1500
!
l2vpn xconnect context A
 member pseudowire 100
 member gigabitethernet 0/0/0.1
```

CE2 Configuration

```
interface fastethernet0/0/0
 no ip address
interface fastethernet0/0/0.1
 encapsulation dot1Q 100
 ip address 10.181.182.2 255.255.255.0
```

The **show l2vpn atom binding** command, issued from router PE1, shows a matching MTU value of 1500 bytes on both the local and remote routers:

```
Device# show l2vpn atom binding
Destination Address: 10.1.1.152,  VC ID: 100
    Local Label: 100
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]
    Remote Label: 202
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: RA [2]
              CV Type: LSPV [2]
```

Example: Configuring Tunnel Selection

The following example shows how to set up two preferred paths for PE1. One preferred path specifies an MPLS traffic engineering tunnel. The other preferred path specifies an IP address of a loopback address on PE2. There is a static route configured on PE1 that uses a TE tunnel to reach the IP address on PE2.

PE1 Configuration

```
mpls label protocol ldp
mpls traffic-eng tunnels
tag-switching tdp router-id Loopback0
pseudowire-class pw1
 encapsulation mpls
 preferred-path interface Tunnel1 disable-fallback
!
pseudowire-class pw2
 encapsulation mpls
 preferred-path peer 10.18.18.18
!
interface Loopback0
 ip address 10.2.2.2 255.255.255.255
 no ip directed-broadcast
 no ip mroute-cache
!
interface Tunnel1
 ip unnumbered Loopback0
 no ip directed-broadcast
 tunnel destination 10.16.16.16
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth 1500
 tunnel mpls traffic-eng path-option 1 explicit name path-tu1
!
interface Tunnel2
 ip unnumbered Loopback0
 no ip directed-broadcast
 tunnel destination 10.16.16.16
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth 1500
 tunnel mpls traffic-eng path-option 1 dynamic
!
interface gigabitethernet0/0/0
 no ip address
 no ip directed-broadcast
 no negotiation auto
!
interface gigabitethernet0/0/0.1
 encapsulation dot1Q 222
 no ip directed-broadcast
 xconnect 10.16.16.16 101 pw-class pw1
!
interface ATM1/0/0
 no ip address
 no ip directed-broadcast
 no atm enable-ilmi-trap
 no atm ilmi-keepalive
 pvc 0/50 l2transport
  encapsulation aal5
  xconnect 10.16.16.16 150 pw-class pw2
!
```

```
interface FastEthernet2/0/1
 ip address 10.0.0.1 255.255.255.0
 no ip directed-broadcast
 tag-switching ip
 mpls traffic-eng tunnels
 ip rsvp bandwidth 15000 15000
!
router ospf 1
 log-adjacency-changes
 network 10.0.0.0 0.0.0.255 area 0
 network 10.2.2.2 0.0.0.0 area 0
 mpls traffic-eng router-id Loopback0
 mpls traffic-eng area 0
!
ip route 10.18.18.18 255.255.255.255 Tunnel2
!
ip explicit-path name path-tu1 enable
 next-address 10.0.0.1
 index 3 next-address 10.0.0.1
```

PE2 Configuration

```
mpls label protocol ldp
mpls traffic-eng tunnels
mpls ldp router-id Loopback0
interface Loopback0
 ip address 10.16.16.16 255.255.255.255
 no ip directed-broadcast
 no ip mroute-cache
!
interface Loopback2
 ip address 10.18.18.18 255.255.255.255
 no ip directed-broadcast
!
interface FastEthernet1/1/0
 ip address 10.0.0.2 255.255.255.0
 no ip directed-broadcast
 mpls traffic-eng tunnels
 mpls ip
 no cdp enable
 ip rsvp bandwidth 15000 15000
!
interface FastEthernet1/1/1
 no ip address
 no ip directed-broadcast
 no cdp enable
!
interface FastEthernet1/1/1.1
 encapsulation dot1Q 222
 no ip directed-broadcast
 no cdp enable
 mpls l2transport route 10.2.2.2 101
!
interface ATM5/0/0
 no ip address
 no ip directed-broadcast
 no atm enable-ilmi-trap
 no atm ilmi-keepalive
 pvc 0/50 l2transport
  encapsulation aal5
  xconnect 10.2.2.2 150 encapsulation mpls
!
router ospf 1
```

```
 log-adjacency-changes
 network 10.0.0.0 0.0.0.255 area 0
 network 10.16.16.16 0.0.0.0 area 0
 mpls traffic-eng router-id Loopback0
 mpls traffic-eng area 0
```

Example: Configuring Tunnel Selection Using Commands Associated with L2VPN Protocol-Based Feature

The following example shows how to set up two preferred paths for PE1. One preferred path specifies an MPLS traffic engineering tunnel. The other preferred path specifies an IP address of a loopback address on PE2. There is a static route configured on PE1 that uses a TE tunnel to reach the IP address on PE2.

PE1 Configuration

```
mpls label protocol ldp
mpls traffic-eng tunnels
tag-switching tdp router-id Loopback0
template type pseudowire pw1
 encapsulation mpls
 preferred-path interface Tunnel1 disable-fallback
!
template type pseudowire pw2
 encapsulation mpls
 preferred-path peer 10.18.18.18
!
interface Loopback0
 ip address 10.2.2.2 255.255.255.255
 no ip directed-broadcast
 no ip mroute-cache
!
interface Tunnel1
 ip unnumbered Loopback0
 no ip directed-broadcast
 tunnel destination 10.16.16.16
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth 1500
 tunnel mpls traffic-eng path-option 1 explicit name path-tu1
!
interface Tunnel2
 ip unnumbered Loopback0
 no ip directed-broadcast
 tunnel destination 10.16.16.16
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth 1500
 tunnel mpls traffic-eng path-option 1 dynamic
!
interface gigabitethernet0/0/0
 no ip address
 no ip directed-broadcast
 no negotiation auto
!
interface gigabitethernet0/0/0.1
 encapsulation dot1Q 222
 no ip directed-broadcast
 interface pseudowire 100
 source template type pseudowire pw1
```

```
  neighbor 10.16.16.16 101
!
l2vpn xconnect context con1
!
interface ATM1/0/0
 no ip address
 no ip directed-broadcast
 no atm enable-ilmi-trap
 no atm ilmi-keepalive
 pvc 0/50 l2transport
  encapsulation aal5
 interface pseudowire 100
 source template type pseudowire pw2
 neighbor 10.16.16.16 150
!
l2vpn xconnect context con1
!
interface FastEthernet2/0/1
 ip address 10.0.0.1 255.255.255.0
 no ip directed-broadcast
 tag-switching ip
 mpls traffic-eng tunnels
 ip rsvp bandwidth 15000 15000
!
router ospf 1
 log-adjacency-changes
 network 10.0.0.0 0.0.0.255 area 0
 network 10.2.2.2 0.0.0.0 area 0
 mpls traffic-eng router-id Loopback0
 mpls traffic-eng area 0
!
ip route 10.18.18.18 255.255.255.255 Tunnel2
!
ip explicit-path name path-tu1 enable
 next-address 10.0.0.1
 index 3 next-address 10.0.0.1
```

PE2 Configuration

```
mpls label protocol ldp
mpls traffic-eng tunnels
mpls ldp router-id Loopback0
interface Loopback0
 ip address 10.16.16.16 255.255.255.255
 no ip directed-broadcast
 no ip mroute-cache
!
interface Loopback2
 ip address 10.18.18.18 255.255.255.255
 no ip directed-broadcast
!
interface FastEthernet1/1/0
 ip address 10.0.0.2 255.255.255.0
 no ip directed-broadcast
 mpls traffic-eng tunnels
 mpls ip
 no cdp enable
 ip rsvp bandwidth 15000 15000
!
interface FastEthernet1/1/1
 no ip address
 no ip directed-broadcast
 no cdp enable
```

```
!
interface FastEthernet1/1/1.1
 encapsulation dot1Q 222
 no ip directed-broadcast
 no cdp enable
 mpls l2transport route 10.2.2.2 101
!
interface ATM5/0/0
 no ip address
 no ip directed-broadcast
 no atm enable-ilmi-trap
 no atm ilmi-keepalive
 pvc 0/50 l2transport
  encapsulation aal5
  interface pseudowire 100
   encapsulation mpls
   neighbor 10.2.2.2 150
!
l2vpn xconnect context A
  member pseudowire 100
  member GigabitEthernet0/0/0.1
!
router ospf 1
 log-adjacency-changes
 network 10.0.0.0 0.0.0.255 area 0
 network 10.16.16.16 0.0.0.0 area 0
 mpls traffic-eng router-id Loopback0
 mpls traffic-eng area 0
```

Example: Configuring MTU Values in xconnect Configuration Mode for L2VPN Interworking

The following example shows an L2VPN Interworking example. The PE1 router has a serial interface configured with an MTU value of 1492 bytes. The PE2 router uses xconnect configuration mode to set a matching MTU of 1492 bytes, which allows the two routers to form an interworking VC. If the PE2 router did not set the MTU value in xconnect configuration mode, the interface would be set to 1500 bytes by default and the VC would not come up.

Note L2VPN interworking is not supported on Cisco ASR 900 RSP3 Module.

PE1 Configuration

```
pseudowire-class atom-ipiw
 encapsulation mpls
 interworking ip
!
interface Loopback0
 ip address 10.1.1.151 255.255.255.255
!
interface Serial2/0/0
 mtu 1492
 no ip address
 encapsulation ppp
 no fair-queue
 serial restart-delay 0
```

```
 xconnect 10.1.1.152 123 pw-class atom-ipiw
!
interface Serial4/0/0
 ip address 10.151.100.1 255.255.255.252
 encapsulation ppp
 mpls ip
 serial restart-delay 0
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.151 0.0.0.0 area 0
 network 10.151.100.0 0.0.0.3 area 0
!
mpls ldp router-id Loopback0
```

PE2 Configuration

```
pseudowire-class atom-ipiw
 encapsulation mpls
 interworking ip
!
interface Loopback0
 ip address 10.1.1.152 255.255.255.255
!
interface FastEthernet0/0/0
 no ip address
 xconnect 10.1.1.151 123 pw-class atom-ipiw
  mtu 1492
!
interface Serial4/0/0
 ip address 10.100.152.2 255.255.255.252
 encapsulation ppp
 mpls ip
 serial restart-delay 0
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.152 0.0.0.0 area 0
 network 10.100.152.0 0.0.0.3 area 0
!
mpls ldp router-id Loopback0
```

The **show mpls l2transport binding** command shows that the MTU value for the local and remote routers is 1492 bytes.

PE1

```
Router# show mpls l2transport binding
Destination Address: 10.1.1.152,  VC ID: 123
    Local Label: 105
        Cbit: 1,    VC Type: PPP,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]
    Remote Label: 205
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: RA [2]
              CV Type: LSPV [2]
Router# show mpls l2transport vc detail
Local interface: Serial2/0/0 up, line protocol up, PPP up
```

```
  MPLS VC type is PPP, interworking type is IP
  Destination address: 10.1.1.152, VC ID: 123, VC status: up
    Output interface: Serial4/0/0, imposed label stack {1003 205}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:25:29, last status change time: 00:24:54
  Signaling protocol: LDP, peer 10.1.1.152:0 up
    Targeted Hello: 10.1.1.151(LDP Id) -> 10.1.1.152
    Status TLV support (local/remote)   : enabled/supported
      Label/status state machine        : established, LruRru
      Last local dataplane   status rcvd: no fault
      Last local SSS circuit status rcvd: no fault
      Last local SSS circuit status sent: no fault
      Last local  LDP TLV    status sent: no fault
      Last remote LDP TLV    status rcvd: no fault
    MPLS VC labels: local 105, remote 205
    Group ID: local n/a, remote 0
    MTU: local 1492, remote 1492
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 30, send 29
    byte totals:   receive 2946, send 3364
    packet drops:  receive 0, send 0
```

PE2

```
Router# show mpls l2transport binding
Destination Address: 10.1.1.151,  VC ID: 123
    Local Label: 205
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: RA [2]
              CV Type: LSPV [2]
    Remote Label: 105
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]
Router# show mpls l2transport vc detail
Local interface: Fe0/0/0 up, line protocol up, FastEthernet up
  MPLS VC type is FastEthernet, interworking type is IP
  Destination address: 10.1.1.151, VC ID: 123, VC status: up
    Output interface: Se4/0/0, imposed label stack {1002 105}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:25:19, last status change time: 00:25:19
  Signaling protocol: LDP, peer 10.1.1.151:0 up
    Targeted Hello: 10.1.1.152(LDP Id) -> 10.1.1.151
    Status TLV support (local/remote)   : enabled/supported
      Label/status state machine        : established, LruRru
      Last local dataplane   status rcvd: no fault
      Last local SSS circuit status rcvd: no fault
      Last local SSS circuit status sent: no fault
      Last local  LDP TLV    status sent: no fault
      Last remote LDP TLV    status rcvd: no fault
    MPLS VC labels: local 205, remote 105
    Group ID: local n/a, remote 0
    MTU: local 1492, remote 1492
    Remote interface description:
  Sequencing: receive disabled, send disabled
```

```
  VC statistics:
    packet totals: receive 29, send 30
    byte totals:   receive 2900, send 3426
    packet drops:  receive 0, send 0
```

Example: Configuring MTU Values in xconnect Configuration Mode for L2VPN Interworking Using Commands Associated with L2VPN Protocol-Based Feature

The following example shows an L2VPN Interworking example. The PE1 router has a serial interface configured with an MTU value of 1492 bytes. The PE2 router uses xconnect configuration mode to set a matching MTU of 1492 bytes, which allows the two routers to form an interworking VC. If the PE2 router did not set the MTU value in xconnect configuration mode, the interface would be set to 1500 bytes by default and the VC would not come up.

PE1 Configuration

```
template type pseudowire atom-ipiw
 encapsulation mpls
 interworking ip
!
interface Loopback0
 ip address 10.1.1.151 255.255.255.255
!
interface Serial2/0/0
 mtu 1492
 no ip address
 encapsulation ppp
 no fair-queue
 serial restart-delay 0
interface pseudowire 100
 source template type pseudowire atom-ipiw
 neighbor 10.1.1.152 123
!
l2vpn xconnect context con1
 member <ac_int>
 member pseudowire 100
!
interface Serial4/0/0
 ip address 10.151.100.1 255.255.255.252
 encapsulation ppp
 mpls ip
 serial restart-delay 0
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.151 0.0.0.0 area 0
 network 10.151.100.0 0.0.0.3 area 0
!
mpls ldp router-id Loopback0
```

PE2 Configuration

```
template type pseudowire atom-ipiw
 encapsulation mpls
 interworking ip
!
interface Loopback0
 ip address 10.1.1.152 255.255.255.255
```

```
!
interface FastEthernet0/0/0
 no ip address
 interface pseudowire 100
 source template type pseudowire atom-ipiw
 neighbor 10.1.1.151 123
!
l2vpn xconnect context con1
 member <ac_int>
 member pseudowire1
!
interface Serial4/0/0
 ip address 10.100.152.2 255.255.255.252
 encapsulation ppp
 mpls ip
 serial restart-delay 0
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.152 0.0.0.0 area 0
 network 10.100.152.0 0.0.0.3 area 0
!
mpls ldp router-id Loopback0
```

The **show l2vpn atom binding** command shows that the MTU value for the local and remote routers is 1492 bytes.

PE1

```
Device# show l2vpn atom binding
Destination Address: 10.1.1.152,  VC ID: 123
    Local Label: 105
        Cbit: 1,    VC Type: PPP,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]
    Remote Label: 205
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: RA [2]
              CV Type: LSPV [2]
Device# show l2vpn atom vc detail
Local interface: Serial2/0/0 up, line protocol up, PPP up
  MPLS VC type is PPP, interworking type is IP
  Destination address: 10.1.1.152, VC ID: 123, VC status: up
    Output interface: Serial4/0/0, imposed label stack {1003 205}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:25:29, last status change time: 00:24:54
  Signaling protocol: LDP, peer 10.1.1.152:0 up
    Targeted Hello: 10.1.1.151(LDP Id) -> 10.1.1.152
    Status TLV support (local/remote)   : enabled/supported
      Label/status state machine        : established, LruRru
      Last local dataplane   status rcvd: no fault
      Last local SSS circuit status rcvd: no fault
      Last local SSS circuit status sent: no fault
      Last local  LDP TLV    status sent: no fault
      Last remote LDP TLV    status rcvd: no fault
    MPLS VC labels: local 105, remote 205
    Group ID: local n/a, remote 0
    MTU: local 1492, remote 1492
    Remote interface description:
```

```
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 30, send 29
    byte totals:   receive 2946, send 3364
    packet drops:  receive 0, send 0
```

PE2

```
Device# show l2vpn atom binding
Destination Address: 10.1.1.151,  VC ID: 123
    Local Label: 205
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: RA [2]
              CV Type: LSPV [2]
    Remote Label: 105
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1492,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]
Device# show l2vpn atom vc detail
Local interface: Fe0/0/0 up, line protocol up, FastEthernet up
  MPLS VC type is FastEthernet, interworking type is IP
  Destination address: 10.1.1.151, VC ID: 123, VC status: up
    Output interface: Se4/0/0, imposed label stack {1002 105}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:25:19, last status change time: 00:25:19
  Signaling protocol: LDP, peer 10.1.1.151:0 up
    Targeted Hello: 10.1.1.152(LDP Id) -> 10.1.1.151
    Status TLV support (local/remote)   : enabled/supported
      Label/status state machine        : established, LruRru
      Last local dataplane   status rcvd: no fault
      Last local SSS circuit status rcvd: no fault
      Last local SSS circuit status sent: no fault
      Last local  LDP TLV    status sent: no fault
      Last remote LDP TLV    status rcvd: no fault
    MPLS VC labels: local 205, remote 105
    Group ID: local n/a, remote 0
    MTU: local 1492, remote 1492
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 29, send 30
    byte totals:   receive 2900, send 3426
    packet drops:  receive 0, send 0
```

Examples: Configuring Any Transport over MPLS (AToM) Remote Ethernet Port Shutdown

The following example shows how to enable remote Ethernet port shutdown:

```
configure terminal
!
pseudowire-class eompls
 encapsulation mpls
!
interface GigabitEthernet1/0/0
```

```
 xconnect 10.1.1.1 1 pw-class eompls
  remote link failure notification
```

The following example shows how to disable remote Ethernet port shutdown:

```
configure terminal
!
pseudowire-class eompls
 encapsulation mpls
!
interface GigabitEthernet1/0/0
 xconnect 10.1.1.1 1 pw-class eompls
  no remote link failure notification
```

The related **show** command output reports operational status for all remote L2 Tunnels by interface.

```
Router# show interface G1/0/0
GigabitEthernet1/0/0 is L2 Tunnel remote down, line protocol is up
Hardware is GigMac 4 Port GigabitEthernet, address is 0003.ff4e.12a8 (bia 0003.ff4e.12a8)
  Internet address is 10.9.9.2/16
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec, rely 255/255, load 1/255
Router# show ip interface brief
Interface              IP-Address      OK? Method Status Protocol
GigabitEthernet2/0/0 unassigned      YES NVRAM  L2 Tunnel remote down up
GigabitEthernet2/1/0 unassigned      YES NVRAM  administratively down down
```

Note Remote Ethernet port shutdown is enabled by default when EVC "default encapsulation" is configured.

Examples: Configuring Any Transport over MPLS (AToM) Remote Ethernet Port Shutdown Using Commands Associated with L2VPN Protocol-Based Feature

The following example shows how to enable remote Ethernet port shutdown:

```
configure terminal
!
template type pseudowire eompls
 encapsulation mpls
!
interface GigabitEthernet1/0/0
interface pseudowire 100
 source template type pseudowire eompls
 neighbor 10.1.1.1 1
!
l2vpn xconnect context con1
 remote link failure notification
```

The following example shows how to disable remote Ethernet port shutdown:

```
configure terminal
!
template type pseudowire eompls
 encapsulation mpls
!
interface GigabitEthernet1/0/0
 interface pseudowire 100
 source template type pseudowire eompls
```

```
 neighbor 10.1.1.1 1
!
l2vpn xconnect context con1
 no remote link failure notification
```

The related **show** command output reports operational status for all remote L2 Tunnels by interface.

```
Router# show interface G1/0/0
GigabitEthernet1/0/0 is L2 Tunnel remote down, line protocol is up
Hardware is GigMac 4 Port GigabitEthernet, address is 0003.ff4e.12a8 (bia 0003.ff4e.12a8)
  Internet address is 10.9.9.2/16
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec, rely 255/255, load 1/255
Router# show ip interface brief
Interface              IP-Address      OK? Method Status Protocol
GigabitEthernet2/0/0 unassigned      YES NVRAM  L2 Tunnel remote down up
GigabitEthernet2/1/0 unassigned      YES NVRAM  administratively down down
```

Additional References for Any Transport over MPLS

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Any Transport over MPLS

ß

Table 10: Feature Information for Any Transport over MPLS

Feature Name	Releases	Feature Information
Any Transport over MPLS (AToM): ATM AAL5 over MPLS (AAL5oMPLS)	Cisco IOS XE Release 3.2S Cisco IOS XE Release 3.6S	In Cisco IOS XE Release 3.2S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS XE Release 3.6S, support was added for the Cisco ASR 903 Router. This feature introduced no new or modified commands.
Any Transport over MPLS (AToM): ATM Cell Relay over MPLS: Packed Cell Relay	Cisco IOS XE Release 3.5S	In Cisco IOS XE Release 3.5S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 Router.
Any Transport over MPLS (AToM): ATM OAM Emulation	Cisco IOS XE Release 3.2S	In Cisco IOS XE Release 3.2S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. This feature introduced no new or modified commands.
	Cisco IOS XE Release 2.5	This feature provides capability to support sequencing of AToM) data plane packets.
Any Transport over MPLS (AToM): Ethernet over MPLS (EoMPLS)	Cisco IOS XE Release 2.4 Cisco IOS XE Release 3.5S	This feature allows you to transport Layer 2 Ethernet VLAN packets from various sources over an MPLS backbone. Ethernet over MPLS extends the usability of the MPLS backbone by enabling it to offer Layer 2 services in addition to already existing Layer 3 services. You can enable the MPLS backbone network to accept Layer 2 VLAN packets by configuring the PE routers at the both ends of the MPLS backbone. In Cisco IOS XE Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Routers. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 Router.
Any Transport over MPLS (AToM): Ethernet over MPLS: Port Mode (EoMPLS)	Cisco IOS XE Release 2.4	Ethernet over MPLS (EoMPLS) is the transport of Ethernet frames across an MPLS core. It transports all frames received on a particular Ethernet or virtual LAN (VLAN) segment, regardless of the destination Media Access Control (MAC) information. It does not perform MAC learning or MAC look up for forwarding packets from the Ethernet interface. Port mode allows a frame coming into an interface to be packed into an MPLS packet and transported over the MPLS backbone to an egress interface. In Cisco IOS XE Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Routers.

Feature Name	Releases	Feature Information
Any Transport over MPLS-Ethernet over MPLS Enhancements: Fast Reroute	Cisco IOS XE Release 2.4	AToM can use MPLS traffic engineering (TE) tunnels with fast reroute (FRR) support. This features enhances FRR functionality for Ethernet over MPLS (EoMPLS). In Cisco IOS XE Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Routers.
Any Transport over MPLS (AToM): Frame Relay over MPLS (FRoMPLS)	Cisco IOS XE Release 3.2.1S	In Cisco IOS XE Release 3.2.1S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. This feature introduced no new or modified commands.
Any Transport over MPLS (AToM): HDLC over MPLS (HDLCoMPLS)	Cisco IOS XE Release 3.2S	In Cisco IOS XE Release 3.2S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. This feature introduced no new or modified commands.
Any Transport over MPLS (AToM): Layer 2 Quality of Service (QoS)	Cisco IOS XE Release 2.3	This feature provides support for quality of service (QoS) features such as traffic policing, traffic shaping, packet marking, and mapping of the packets. In Cisco IOS XE Release 2.3, this feature was introduced on the Cisco ASR 1000 Series Routers.
Any Transport over MPLS (AToM): PPP over MPLS (PPPoMPLS)	Cisco IOS XE Release 3.2S	In Cisco IOS XE Release 3.2S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. This feature introduced no new or modified commands.
Any Transport over MPLS (AToM): Remote Ethernet Port Shutdown	Cisco IOS XE Release 2.4	This feature allows a service provider edge (PE) router on the local end of an Ethernet over MPLS (EoMPLS) pseudowire to detect a remote link failure and cause the shutdown of the Ethernet port on the local customer edge (CE) router. Because the Ethernet port on the local CE router is shut down, the router does not lose data by continuously sending traffic to the failed remote link. This is beneficial if the link is configured as a static IP route. In Cisco IOS XE Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Routers.
ATM Port Mode Packed Cell Relay over MPLS	Cisco IOS XE Release 3.5S	In Cisco IOS XE Release 3.5S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers.
ATM VC Class Support	Cisco IOS XE Release 2.3	The ATM VC Class Support feature allows you to specify AAL5 and AAL0 encapsulations as part of a VC class. In Cisco IOS XE Release 2.3, this feature was introduced on the Cisco ASR 1000 Series Routers.

Feature Name	Releases	Feature Information
AToM Tunnel Selection	Cisco IOS XE Release 2.3	The AToM Tunnel Selection feature allows you to specify the path that traffic uses. You can specify either an MPLS TE tunnel or destination IP address or domain name server (DNS) name. You also have the option of specifying whether the VCs should use the default path (the path LDP uses for signaling) if the preferred path is unreachable. This option is enabled by default; you must explicitly disable it. In Cisco IOS XE Release 2.3, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers.
AToM: ATM Cell Relay over MPLS: VP Mode	Cisco IOS XE Release 2.3	The AToM: ATM Cell Relay over MPLS: VP Mode feature allows you to insert one ATM cell in each MPLS packet in VP mode. In Cisco IOS XE Release 2.3, this feature was introduced on the Cisco ASR 1000 Series Routers.
AToM: Single Cell Relay-VC Mode	Cisco IOS XE Release 2.3	The AToM Single Cell Relay-VC Mode feature allows you to insert one ATM cell in each MPLS packet in VC mode. In Cisco IOS XE Release 2.3, this feature was introduced on the Cisco ASR 1000 Series Routers.
MPLS MTU Command for GRE Tunnels	Cisco IOS XE Release 2.6	This feature allows you to set the MPLS MTU size in GRE tunnels to the maximum size besides the current default size. The following command was modified for this release: **mpls mtu**.
MPLS L2VPN Clear Xconnect Command	Cisco IOS XE Release 3.1S	These features enable you to: • Reset a VC associated with an interface, a peer address, or on all the configured xconnect circuit attachments • Set the control word on dynamic pseudowires (L2VPN pseudowire control word configuration • Enable ATM cell packing for static pseudowires. The following commands were introduced or modified by these features: **cell-packing**, **clear xconnect**, **control-word**, **encapsulation**(Any Transport over MPLS), **oam-ac emulation-enable**.

Feature Name	Releases	Feature Information
Per-Subinterface MTU for Ethernet over MPLS (EoMPLS)	Cisco IOS XE Release 2.4	This feature provides you with the ability to specify maximum transmission unit (MTU) values in xconnect subinterface configuration mode. When you use xconnect subinterface configuration mode to set the MTU value, you establish a pseudowire connection for situations where the interfaces have different MTU values that cannot be changed. In Cisco IOS XE Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. No commands were new or modified for this release.
VLAN ID Rewrite	Cisco IOS XE Release 2.4	The VLAN ID rewrite feature enables you to use VLAN interfaces with different VLAN IDs at both ends of the tunnel. In Cisco IOS XE Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Routers.
AToM Load Balancing with Single PW	Cisco IOS XE Release 3.4S	The AToM Load Balancing with Single PW feature enables load balancing for packets within the same pseudowire by further classifying packets within the same pseudowire into different flows based on some field in the packet received on attachment circuit. In Cisco IOS XE Release 3.4S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers.
Flow-Aware Transport of MPLS Pseudowires	Cisco IOS XE Release 3.11S	The Flow-Aware Transport of MPLS Pseudowires feature enables load balancing of packets within the same pseudowire by further classifying the packets into different flows by adding a flow label at the bottom of the MPLS label stack.
EoMPLS over IPv6 GRE Tunnel	Cisco IOS XE Release 3.15S	The EoMPLS over IPv6 GRE Tunnel feature supports tunneling of EoMPLS traffic via an IPv6 network by using GRE tunnels.

CHAPTER 4

L2VPN Interworking

Interworking is a transforming function that is required to interconnect two heterogeneous attachment circuits (ACs). Several types of interworking functions exist. The function that is used would depend on the type of ACs being used, the type of data being carried, and the level of functionality required. The two main Layer 2 Virtual Private Network (L2VPN) interworking functions supported in Cisco IOS XE software are bridged and routed interworking.

Layer 2 (L2) transport over multiprotocol label switching (MPLS) and IP already exists for like-to-like ACs, such as Ethernet-to-Ethernet or Point-to-Point Protocol (PPP)-to-PPP. L2VPN Interworking builds on this functionality by allowing disparate ACs to be connected. An interworking function facilitates the translation between different L2 encapsulations.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for L2VPN Interworking

Before you configure L2VPN interworking on a device you must enable Cisco Express Forwarding.

HDLC-to-Ethernet Interworking

- Ensure that the serial controller and interface on the High-Level Data Link Control (HDLC) customer edge (CE) and provider edge (PE) devices are configured.

```
enable
 configure terminal
  controller e1 2/0
   channel-group 0 timeslots 1
   no shutdown
!
interface Serial 2/0:0
 no shutdown
end
```

- Before configuring HDLC-to-Ethernet bridged interworking, ensure that bridging is configured on the HDLC CE device.

```
enable
 configure terminal
  bridge irb
  bridge 1 protocol ieee
  bridge 1 route ip
!
interface Serial 2/0:0
 no bridge-group 1
 no ip address
!
interface BVI1
 no ip address
 ip address 192.0.2.1 255.255.255.0
 no shutdown
!
interface Serial 2/0:0
 no ip address
 encapsulation hdlc
 bridge-group 1
  no shutdown
end
```

- Before configuring HDLC-to-Ethernet routed interworking, ensure that an IP address is configured on the HDLC CE device.

```
interface Serial 2/0:0
 ip address 192.0.2.1 255.255.255.0
 encapsulation hdlc
 no shutdown
end
```

Restrictions for L2VPN Interworking

General Restrictions for L2VPN Interworking

This section lists general restrictions that apply to L2VPN interworking. Other restrictions that are platform-specific or device-specific are listed in the following sections.

- MTU configured on the AC should not exceed the MTU in the core of the network because fragmentation is not supported.

- The interworking type on one provider edge (PE) router must match the interworking type on the peer PE router.
- IP interworking with native VLANs is not supported.
- Ethernet VLAN (Type 4) interworking is not supported.
- Only the following Quality of Service (QoS) features are supported with L2VPN interworking:
    - Static IP type of service (ToS) or MPLS experimental bit (EXP) setting in tunnel header.
    - One-to-one mapping of VLAN priority bits to MPLS EXP bits.
- VRF-aware Layer 2 Tunneling Protocol Version 3 (L2TPv3) is not supported on Cisco ASR 1000 platforms.

Restrictions for Routed Interworking

Routed interworking has the following restrictions:

- Multipoint Frame Relay (FR) is not supported.
- QoS classification on IP ToS, DSCP and other IP header fields is not supported.
- Security access control list (ACL) and other features based on IP header fields parsing are not supported.
- In routed mode, only one customer edge (CE) router can be attached to an Ethernet PE router.
- There must be a one-to-one relationship between an AC and the pseudowire. Point-to-multipoint or multipoint-to-point configurations are not supported.
- You must configure routing protocols for point-to-point operation on the CE routers when configuring an Ethernet to non-Ethernet setup.
- In the IP interworking mode, the IPv4 (0800) translation is supported. The PE router captures Address Resolution Protocol (ARP) (0806) packets and responds with its own MAC address (proxy ARP). Everything else is dropped.
- The Ethernet must contain only two IP devices: PE router and CE router. The PE router performs proxy ARP and responds to all ARP requests it receives. Therefore, only one CE router and one PE router should be on the Ethernet segment.
- If the CE routers are doing static routing, you can perform the following tasks:
    - The PE router needs to learn the MAC address of the CE router to correctly forward traffic to it. The Ethernet PE router sends an Internet Control Message Protocol (ICMP) Router Discovery Protocol (RDP) solicitation message with the source IP address as zero. The Ethernet CE router responds to this solicitation message. To configure the Cisco CE router's Ethernet interface to respond to the ICMP RDP solicitation message, issue the **ip irdp** command in interface configuration mode. If you do not configure the CE router, traffic is dropped until the CE router sends traffic toward the PE router.
    - To disable the CE routers from running the router discovery protocol, issue the **ip irdp maxadvertinterval 0** command in interface configuration mode.

- When you change the interworking configuration on an Ethernet PE router, clear the ARP entry on the adjacent CE router so that it can learn the new MAC address. Otherwise, you might experience traffic drops.

Restrictions for PPP Interworking

The following restrictions apply to PPP interworking:

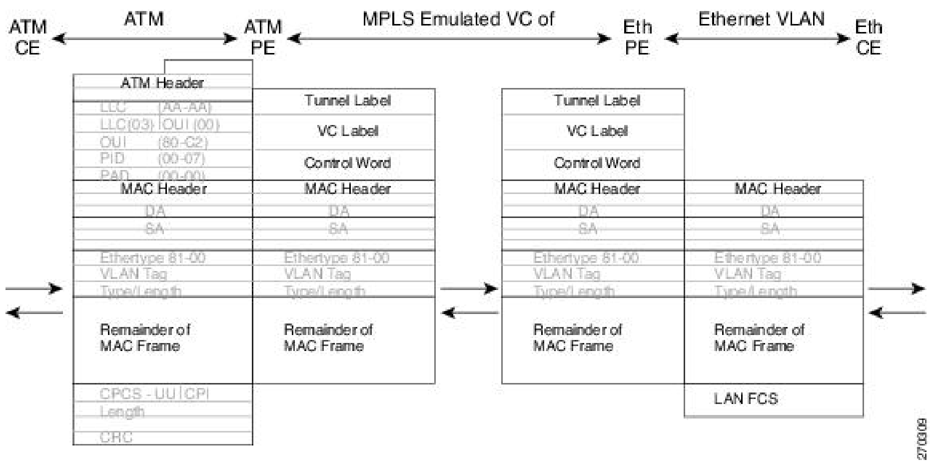
- There must be a one-to-one relationship between a PPP session and the pseudowire. Multiplexing of multiple PPP sessions over the pseudowire is not supported.
- Only IP (IPv4 (0021) interworking is supported. Link Control Protocol (LCP) packets and Internet Protocol Control Protocol (IPCP) packets are terminated at the PE router. Everything else is dropped.
- By default, the PE router assumes that the CE router knows the remote CE router's IP address.
- Password Authentication Protocol (PAP) and Challenge-Handshake Authentication Protocol (CHAP) authentication are supported.

Restrictions for Ethernet/VLAN-to-ATM AAL5 Interworking

The Ethernet/VLAN to ATM AAL5 Any Transport over MPLS (AToM) has the following restrictions:

- Only the following translations are supported; other translations are dropped:
  - Ethernet without LAN FCS (AAAA030080C200070000)
  - Spanning tree (AAAA030080C2000E)
- The ATM encapsulation type supported for bridged interworking is aal5snap. However, ATM encapsulation types supported for routed interworking are aal5snap and aal5mux.
- The existing QoS functionality for ATM is supported, including setting the ATM CLP bit.
- Only ATM AAL5 VC mode is supported. ATM VP and port mode are not supported.
- SVCs are not supported.
- Individual AAL5 ATM cells are assembled into frames before being sent across the pseudowire.
- Non-AAL5 traffic, (such as Operation, Administration, and Maintenance (OAM) cells) is punted to be processed at the route processor (RP) level. A VC that has been configured with OAM cell emulation on the ATM PE router (using the **oam-ac emulation-enable** CLI command) can send end-to-end F5 loopback cells at configured intervals toward the CE router.
- When the pseudowire is down, an F5 end-to-end segment alarm indication signal/remote defect indication (AIS/RDI) is sent from the PE router to the CE router.
- If the Ethernet frame arriving from the Ethernet CE router includes a 802.1Q header (VLAN header), due to the type of endpoint attachment (Ethernet port mode), the VLAN header stays in the frame across the pseudowire (see the figure below).

Figure 9: Protocol Stack for ATM-to-Ethernet AToM Bridged Interworking--with VLAN Header

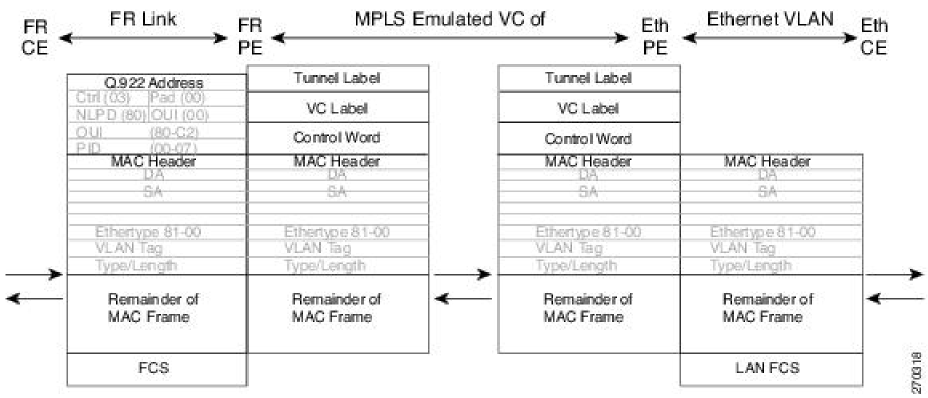


Restrictions for Ethernet/VLAN-to-Frame Relay Interworking

The Ethernet/VLAN-to-Frame Relay AToM has the following restrictions:

- Only the following translations are supported; other translations are dropped:
  - Ethernet without LAN FCS (0300800080C20007)
  - Spanning tree (0300800080C2000E)
- The PE router automatically supports translation of both Cisco and IETF Frame Relay encapsulation types coming from the CE router, but translates only to IETF when sending to the CE router. This is not a problem for the Cisco CE router, because it can manage IETF encapsulation upon receipt even if it is configured to send a Cisco encapsulation.
- The PVC status signaling works the same way as in the like-to-like case. The PE router reports the PVC status to the CE router based upon the availability of the pseudowire.
- The AC maximum transmission unit (MTU) must be within the supported range of MTUs when connected over MPLS.
- Only Frame Relay DLCI mode is supported. Frame Relay port mode is not supported.
- If the Ethernet frame includes a 802.1Q header (VLAN header), due to the type of endpoint attachment (Ethernet port mode), the VLAN header stays in the frame across the pseudowire (see the figure below).
- Frame Relay encapsulation types supported for routed interworking are Cisco and IETF for incoming traffic. However, IETF is also supported for outgoing traffic traveling to the CE router.

Figure 10: Protocol Stack for Frame Relay-to-Ethernet AToM Bridged Interworking--with VLAN Header



Restrictions for HDLC-to-Ethernet Interworking

- The “none CISCO” High-Level Data Link Control (HDLC) encapsulation is not supported.
- IPv6 is not supported in routed mode.

Information About L2VPN Interworking

Overview of L2VPN Interworking

L2 transport over MPLS and IP already exists for like-to-like ACs, such as Ethernet-to-Ethernet or PPP-to-PPP. L2VPN Interworking builds on this functionality by allowing disparate ACs to be connected. An interworking function facilitates the translation between the different L2 encapsulations.

Only the following interworking combinations are supported:

- ATM-to-Ethernet - Routed interworking
- ATM-to-Ethernet - Bridged interworking
- Frame relay-to-Ethernet - Bridged interworking
- PPP-to-Ethernet - Routed interworking
- HDLC-to-Ethernet - Bridged and Routed interworking

L2VPN Interworking Modes

L2VPN interworking works in either Ethernet (bridged) mode or IP (routed) mode. L2VPN interworking does not support Ethernet VLAN (Type 4) mode. You specify the mode in the following ways:

- If using the older legacy CLI commands, you can use the **interworking** {**ethernet** | **ip**} command in pseudowire-class configuration mode.

- If using the newer L2VPN protocol-based CLI commands, you can use the **interworking** {**ethernet** | **ip**} command in xconnect configuration mode.

The **interworking** command causes the ACs to be terminated locally. The two keywords perform the following functions:

- The **ethernet** keyword causes Ethernet frames to be extracted from the AC and sent over the pseudowire. Ethernet end-to-end transmission is resumed. AC frames that are not Ethernet are dropped. In the case of VLAN, the VLAN tag is removed, leaving an untagged Ethernet frame.
- The **ip** keyword causes IP packets to be extracted from the AC and sent over the pseudowire. AC frames that do not contain IPv4 packets are dropped.

The following sections explain more about Ethernet and IP interworking modes.

Ethernet or Bridged Interworking

Ethernet interworking is also called bridged interworking. Ethernet frames are bridged across the pseudowire. The CE routers could be natively bridging Ethernet or could be routing using a bridged encapsulation model, such as Bridge Virtual Interface (BVI) or Routed Bridge Encapsulation (RBE). The PE routers operate in Ethernet like-to-like mode.

This mode is used to offer the following services:

- LAN services--An example is an enterprise that has several sites, where some sites have Ethernet connectivity to the service provider (SP) network and others have ATM connectivity. If the enterprise wants LAN connectivity to all its sites, traffic from the Ethernet or VLAN of one site can be sent through the IP/MPLS network and encapsulated as bridged traffic over an ATM VC of another site.
- Connectivity services--An example is an enterprise that has different sites that are running an Internal Gateway Protocol (IGP) routing protocol, which has incompatible procedures on broadcast and nonbroadcast links. The enterprise has several sites that are running an IGP, such as Open Shortest Path First (OSPF) or Intermediate System-to-Intermediate System (IS-IS), between the sites. In this scenario, some of the procedures (such as route advertisement or designated router) depend on the underlying L2 protocol and are different for a point-to-point ATM connection versus a broadcast Ethernet connection. Therefore, the bridged encapsulation over ATM can be used to achieve homogenous Ethernet connectivity between the CE routers running the IGP.

IP or Routed Interworking

IP interworking is also called routed interworking. The CE routers encapsulate the IP on the link between the CE router and PE router. A new VC type is used to signal the IP pseudowire in MPLS. Translation between the L2 and IP encapsulations across the pseudowire is required. Special consideration needs to be given to the address resolution and routing protocol operation, because these are handled differently on different L2 encapsulations.

This mode is used to provide IP connectivity between sites, regardless of the L2 connectivity to these sites. It is different from a Layer 3 VPN because it is point-to-point in nature and the service provider does not maintain any customer routing information.

Address resolution is encapsulation dependent:

- Ethernet uses Address Resolution Protocol (ARP)
- ATM uses inverse ARP

- PPP uses IP Control Protocol (IPCP)
- HDLC uses Serial Line ARP (SLARP)

Therefore, address resolution must be terminated on the PE router. End-to-end address resolution is not supported. Routing protocols operate differently over broadcast and point-to-point media. For Ethernet, the CE routers must either use static routing or configure the routing protocols to treat the Ethernet side as a point-to-point network.

In routed interworking, IP packets that are extracted from the ACs are sent over the pseudowire. The pseudowire works in the IP Layer 2 transport (VC type 0x000B) like-to-like mode. The interworking function at network service provider's (NSP) end performs the required adaptation based on the AC technology. Non-IPv4 packets are dropped.

In routed interworking, the following considerations are to be kept in mind:

- Address resolution packets (ARP), inverse ARP, and IPCP are punted to the routing protocol. Therefore, NSP at the PE router must provide the following functionality for address resolution:
  - Ethernet--PE device acts as a proxy-ARP server to all ARP requests from the CE router. The PE router responds with the MAC address of its local interface.
  - ATM and Frame Relay point-to-point--By default, inverse ARP does not run in the point-to-point Frame Relay or ATM subinterfaces. The IP address and subnet mask define the connected prefix; therefore, configuration is not required in the CE devices.
- Interworking requires that the MTUs in both ACs match for the pseudowire to come up. The default MTU in one AC should match with the MTU of other AC. The table below lists the range of MTUs that can be configured for different ACs.

Table 11: Range of MTUs for Different ACs

AC type	Range of MTUs supported
ATM	64 to 17940
Gigabit Ethernet	1500 to 4470
POS	64to 9102
Fast Ethernet	64to 9192

Note The MTU configured on the AC should not exceed the MTU in the core network. This ensures that the traffic is not fragmented.

- The CE routers with Ethernet attachment VCs running OSPF must be configured with the **ospfIfType**option so that the OSPF protocol treats the underlying physical broadcast link as a P2P link.

Ethernet VLAN-to-ATM AAL5 Interworking

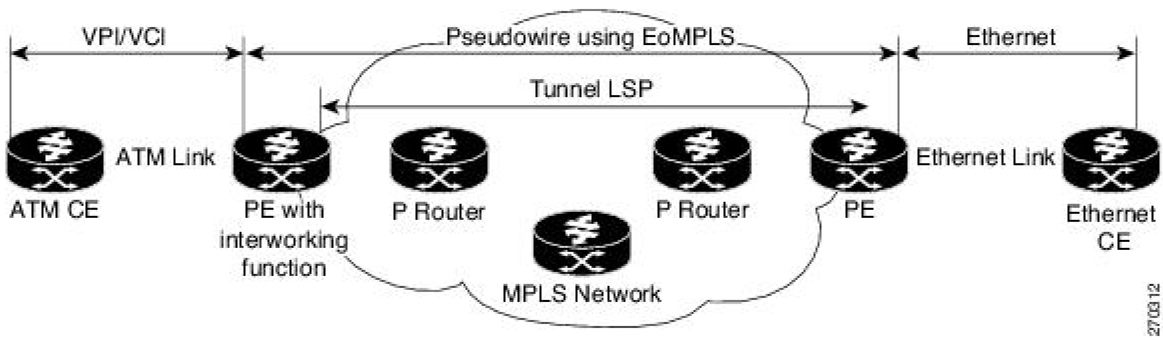
The following topics are covered in this section:

ATM AAL5-to-Ethernet Port AToM--Bridged Interworking

This interworking type provides interoperability between the ATM attachment VC and Ethernet attachment VC connected to different PE routers. Bridged encapsulation corresponding to the bridged (Ethernet) interworking mechanism is used.

The interworking function is performed at the PE router connected to the ATM attachment VC based on multiprotocol encapsulation over ATM AAL5 (see the figure below).

Figure 11: Network Topology for ATM-to-Ethernet AToM Bridged Interworking



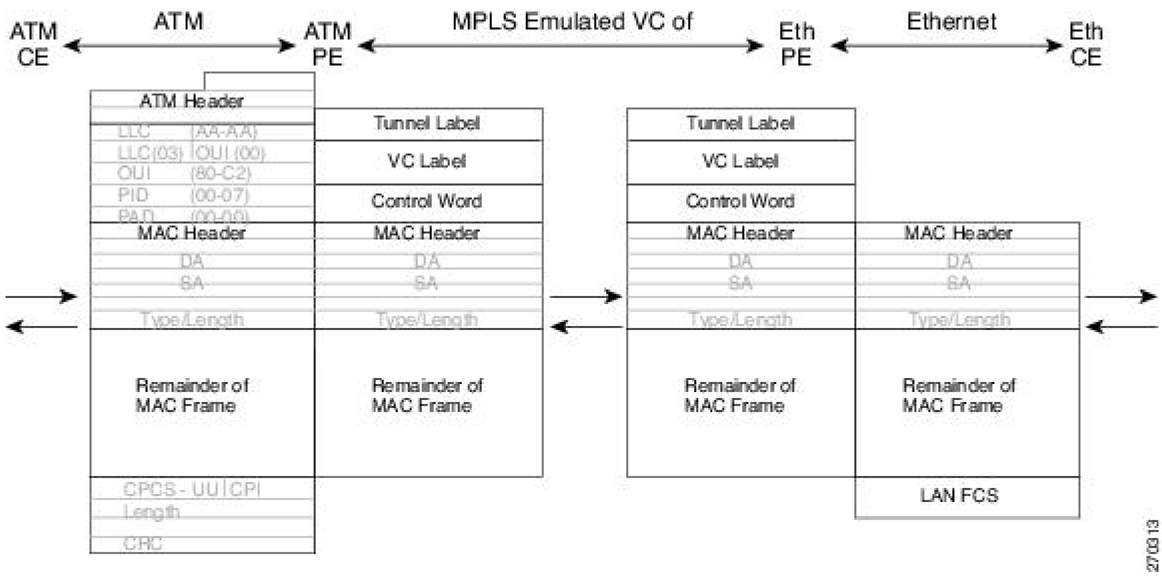
The advantage of this architecture is that the Ethernet PE router (connected to the Ethernet segment) operates similarly to Ethernet like-to-like services.

On the PE router with interworking function, in the direction from the ATM segment to MPLS cloud, the bridged encapsulation (ATM/subnetwork access protocol (SNAP) header) is discarded and the Ethernet frame is encapsulated with the labels required to go through the pseudowire using the VC type 5 (Ethernet) (see the figure below).

In the opposite direction, after the label disposition from the MPLS cloud, Ethernet frames are encapsulated over AAL5 using bridged encapsulation.

The figure below shows the protocol stack for ATM-to-Ethernet AToM bridged interworking. The ATM side has an encapsulation type of aal5snap.

Figure 12: Protocol Stack for ATM-to-Ethernet AToM Bridged Interworking--without VLAN Header



ATM AAL5-to-Ethernet VLAN 802.1Q AToM--Bridged Interworking

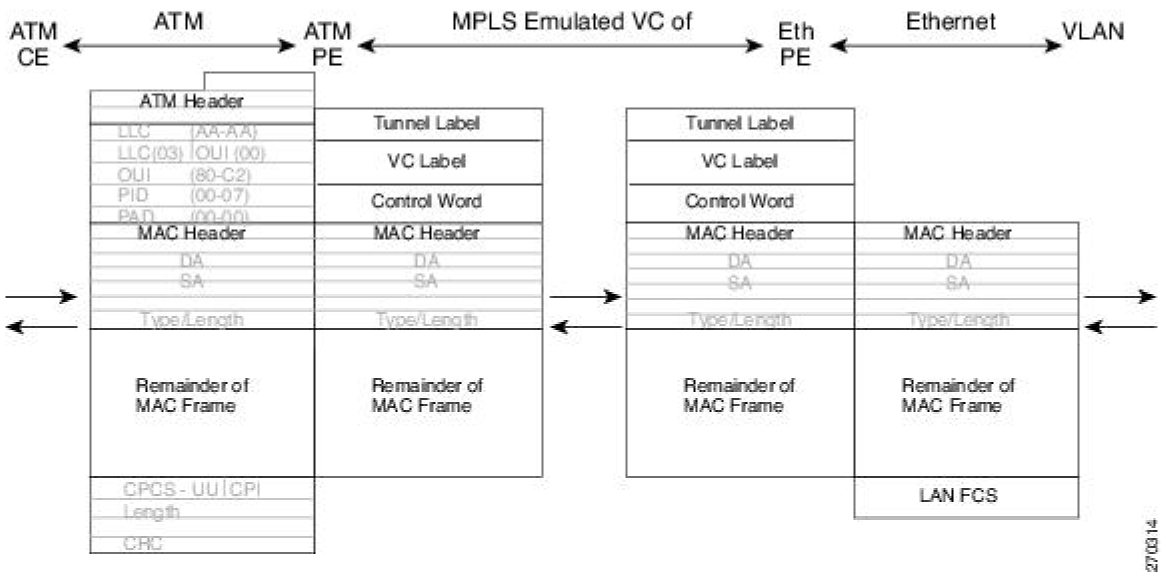
This interworking type provides interoperability between the ATM attachment VC and Ethernet VLAN attachment VC connected to different PE routers. Bridged encapsulation corresponding to the bridged (Ethernet) interworking mechanism is used.

The interworking function is performed in the same way as for the ATM-to-Ethernet port case, implemented on the PE router connected to the ATM attachment VC. The implementation is based on multiprotocol encapsulation over ATM AAL5 (see the figure below).

For the PE router connected to the Ethernet side, one major difference exists due the existence of the VLAN header in the incoming packet. The PE router discards the VLAN header of the incoming frames from the VLAN CE router, and the PE router inserts a VLAN header into the Ethernet frames traveling from the MPLS cloud. The frames sent on the pseudowire (with VC type 5) are Ethernet frames without the VLAN header.

Encapsulation over ATM AAL5 is shown in the figure below.

Figure 13: Protocol Stack for ATM -to-VLAN AToM Bridged Interworking



ATM-to-Ethernet--Routed Interworking

To perform routed interworking, both the ATM PE router and Ethernet PE router must be configured. The figure below shows the routed interworking between ATM to Ethernet. The IP encapsulation over the pseudowire is performed on the ATM packets arriving from the ATM CE router.

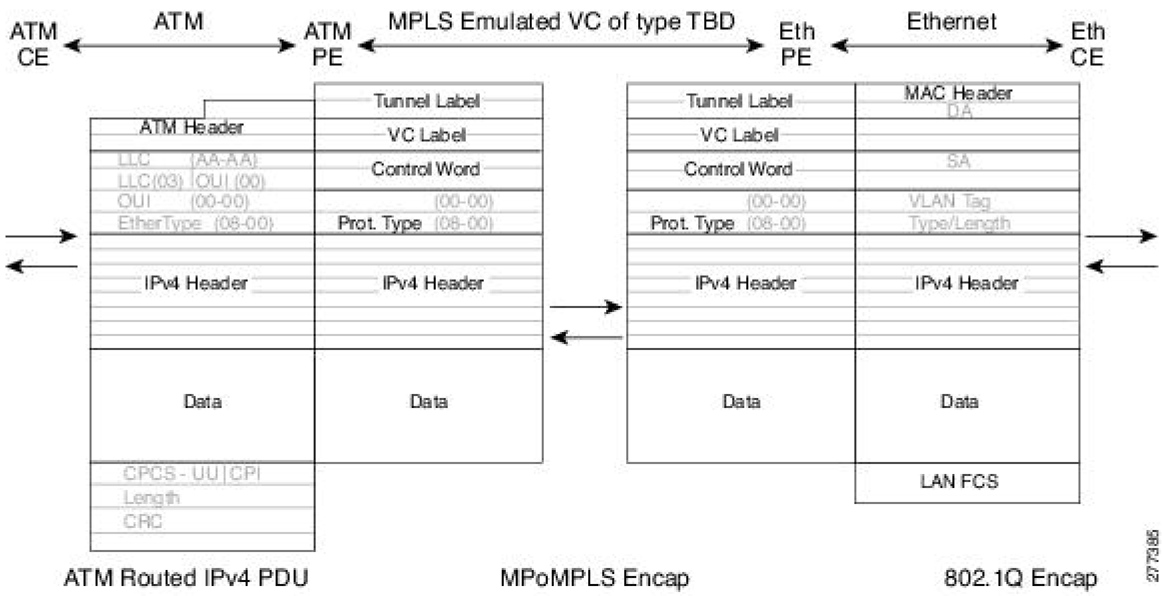
The address resolution is done at the ATM PE router; it is required when the ATM CE router does an inverse ARP. It is not required when the ATM CE router is configured using Point-to-Point (P2P) subinterfaces or static maps.

When packets arrive from the Ethernet CE router, the Ethernet PE router removes the L2 frame tag, and then forwards the IP packet to the egress PE router, using IPoMPLS encapsulation over the pseudowire. The Ethernet PE router makes the forwarding decision based on the L2 circuit ID, the VLAN ID, or port ID, of the incoming L2 frame. At the ATM PE router, after label disposition, the IP packets are encapsulated over the AAL5 using routed encapsulation based on RFC 2684.

The address resolution at the Ethernet PE router can be done when the Ethernet CE router configures the static ARP, or by the proxy ARP on the Ethernet PE router. If the proxy ARP is used, the IP address of the remote CE router can be learned dynamically.

Routing protocols need to be configured to operate in the P2P mode on the Ethernet CE router.

Figure 14: Protocol Stack for ATM-to-Ethernet--Routed Interworking



Ethernet VLAN-to-Frame Relay Interworking

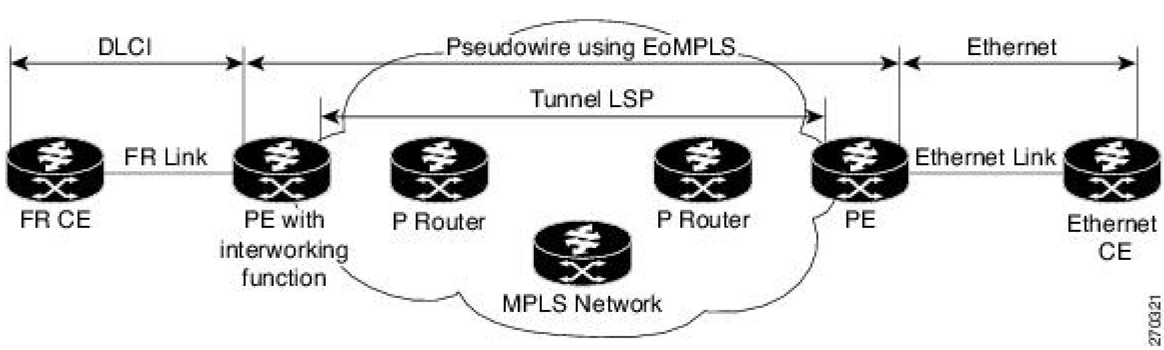
The following topics are covered in this section:

Frame Relay DLCI-to-Ethernet Port AToM--Bridged Interworking

This interworking type provides interoperability between the Frame Relay attachment VC and Ethernet attachment VC connected to different PE routers. Bridged encapsulation corresponding to the bridged (Ethernet) interworking mechanism is used.

For an FR-to-Ethernet port case, the interworking function is performed at the PE router connected to the FR attachment VC based on multiprotocol interconnect over Frame Relay (see the figure below). The interworking is implemented similar to an ATM-to-Ethernet case.

Figure 15: Network Topology for FR-to-Ethernet AToM Bridged Interworking



The advantage of this architecture is that the Ethernet PE router (connected to the Ethernet segment) operates similar to Ethernet like-to-like services: a pseudowire label is assigned to the Ethernet port and then the remote Label Distribution Protocol (LDP) session distributes the labels to its peer PE router. Ethernet frames are carried through the MPLS network using Ethernet over MPLS (EoMPLS).

On the PE router with interworking function, in the direction from the Frame Relay segment to the MPLS cloud, the bridged encapsulation (FR/SNAP header) is discarded and the Ethernet frame is encapsulated with the labels required to go through the pseudowire using the VC type 5 (Ethernet) (see the figure below).

In the opposite direction, after the label disposition from the MPLS cloud, Ethernet frames are encapsulated over Frame Relay using bridged encapsulation.

The following translations are supported:

- Ethernet without LAN FCS (0300800080C20007)
- Spanning tree (0300800080C2000E)

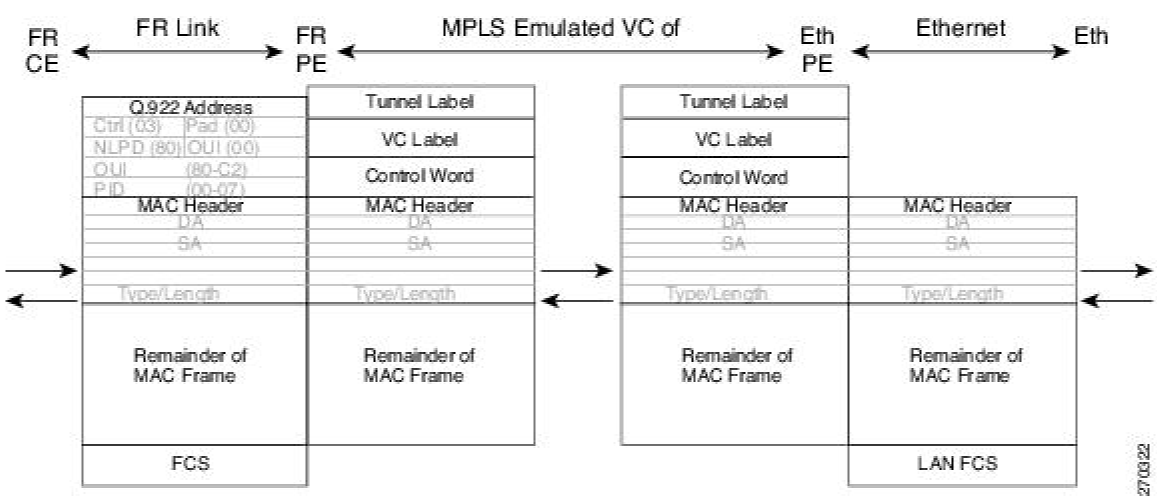
The PE router automatically supports translation of both Cisco and IETF Frame Relay encapsulation types coming from the CE, but translates only to IETF when sending to the CE router. This is not a problem for the Cisco CE router, because it can handle IETF encapsulation on receipt even if it is configured to send Cisco encapsulation.

The existing QoS functionality for Frame Relay is supported. The PVC status signaling works the same way as in the like-to-like case. The PE router reports the PVC status to the CE router, based on the availability of the pseudo wire.

The AC MTU must match when connected over MPLS. Only Frame Relay DLCI mode is supported; Frame Relay port mode is not supported in the bridged interworking.

The figure below shows the protocol stack for FR-to-Ethernet bridged interworking.

Figure 16: Protocol Stack for FR-to-Ethernet AToM Bridged Interworking--without VLAN Header



Frame Relay DLCI-to-Ethernet VLAN 802.1Q AToM--Bridged Interworking

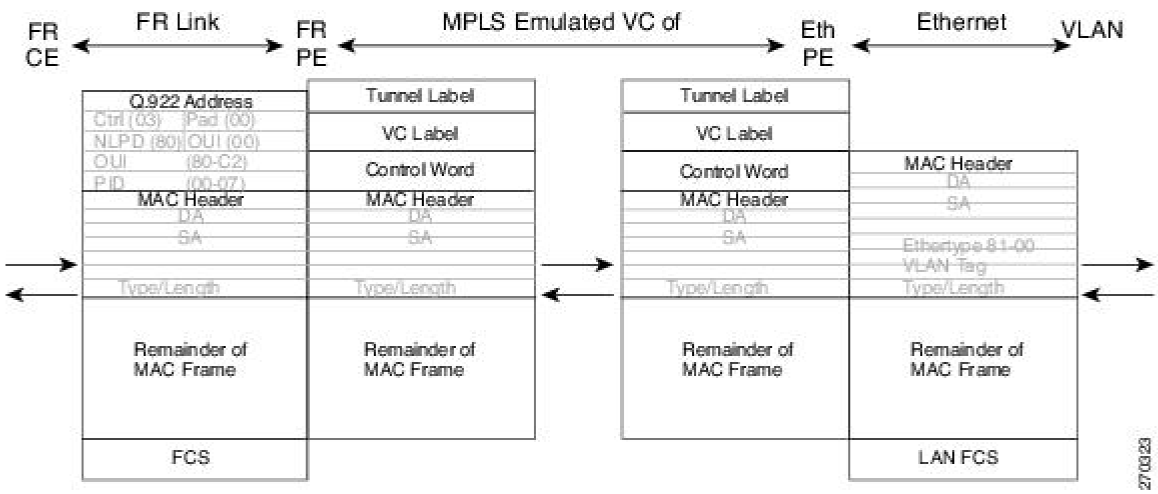
This interworking type provides interoperability between the Frame Relay attachment VC and Ethernet VLAN Attachment VC connected to different PE routers. The bridged encapsulation corresponding to the bridged (Ethernet) interworking mechanism is used.

The interworking function is performed in the same way as it is done for the Frame Relay to Ethernet port case; it is implemented on the PE router connected to the Frame Relay attachment VC, based upon a multiprotocol interconnect over Frame Relay (see the figure above).

As in the ATM-to-VLAN case, one difference exists on the Ethernet side due the existence of the VLAN header in the incoming packet. The PE router on the VLAN side discards the VLAN header of the incoming frames from the VLAN CE router, and the PE router inserts a VLAN header into the Ethernet frames traveling from the MPLS cloud. The frames sent on the pseudowire (with VC type 5) are Ethernet frames without the VLAN header.

The figure below shows the protocol stack for FR-to-VLAN AToM bridged interworking.

Figure 17: Protocol Stack for FR-to-VLAN AToM Bridged Interworking



Frame Relay DLCI-to-Ethernet VLAN Qot1Q QinQ AToM - Bridged Interworking

This interworking type provides interoperability between the Frame Relay Attachment VC and Ethernet VLAN Attachment VC connected to different PE routers. The bridged encapsulation corresponding to bridged (Ethernet) interworking mechanism is used.

The interworking function is done in the same way as it is done for FR-to-Ethernet port case; it is implemented on the PE router connected to the Frame Relay attachment VC, based on RFC 2427(Multiprotocol Interconnect over Frame Relay).

When compared with Frame Relay DLCI-to-Ethernet port AToM, there is one major difference on the Ethernet access side, due the existence of the VLAN header in the incoming packet. The PE router on the VLAN side will discard the VLAN header of the incoming frames form the VLAN CE router, and it will insert a VLAN header into the Ethernet frames coming from the MPLS cloud. So the frames sent on the pseudo wire (with VC type 5) will be Ethernet frames without the VLAN header.

The following translations are supported on the Frame Relay PE router:

- Ethernet without LAN FCS (0300800080C20007)
- Spanning tree (0300800080C2000E)

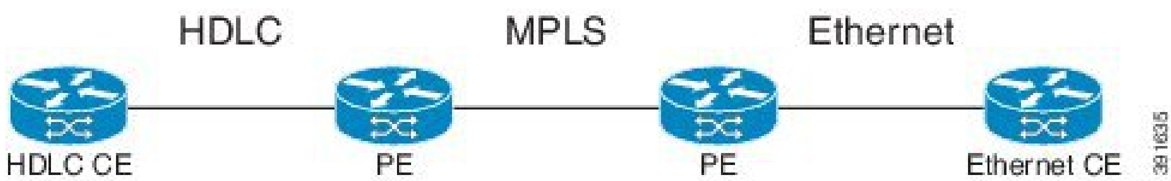
Frame Relay encapsulation types supported for bridged interworking: Cisco and IETF for incoming traffic, IETF only for outgoing traffic towards CE router.

HDLC-to-Ethernet Interworking

High-Level Data Link Control (HDLC) and Ethernet are two independent data link layer transport protocols that utilize the Any Transport over MPLS (AToM) framework to communicate with each other. The interworking function enables translation between two heterogeneous Layer 2 encapsulations over a Multiprotocol Label Switching (MPLS) backbone.

The figure below depicts a simple HDLC-to-Ethernet interworking topology.

Figure 18: HDLC-to-Ethernet interworking topology



HDLC-to-Ethernet interworking supports the following:

- Ethernet or bridged interworking
- IP or routed interworking
- HDLC encapsulation type: CISCO
- Ethernet encapsulation types: IEEE 802.1Q, QinQ, port mode

The HDLC pass-through feature is not affected in any way by HDLC-to-Ethernet interworking.

HDLC-to-Ethernet interworking supports two interworking modes:

- HDLC-to-Ethernet — Ethernet or Bridged interworking
- HDLC-to-Ethernet — IP or Routed interworking

HDLC-to-Ethernet — Ethernet or Bridged Interworking

HDLC-to-Ethernet bridged interworking provides interoperability between the HDLC attachment virtual circuit (VC) and Ethernet VLAN attachment VC connected to different provider edge (PE) devices. Bridged encapsulation corresponding to the bridged (Ethernet) interworking mechanism is used.

When packets arrive from the HDLC customer edge (CE) device, they consist of the HDLC header, the Ethernet MAC header, and the payload. At the HDLC PE device, the HDLC header is removed, and MPLS labels are inserted. The frames are then routed over the pseudowire to the Ethernet PE device, where the MPLS labels are removed. On the Ethernet side, there are two possibilities. The attachment circuit (AC) is either Ethernet or VLAN.

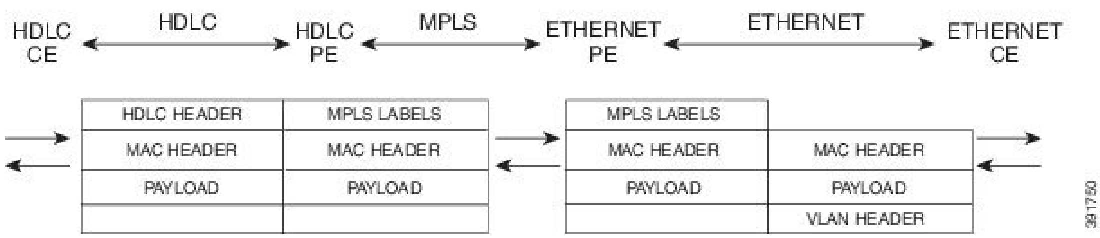
For an Ethernet attachment circuit (AC), the packets are forwarded to the Ethernet CE device, as is. For a VLAN AC, VLAN headers are added at the VLAN/QinQ subinterface's AC. The Ethernet VLAN frame is then forwarded to the VLAN CE device.

In the opposite direction (Ethernet / VLAN to HDLC), the VLAN header is present in the incoming packet, if the AC is VLAN. So, when packets arrive from the VLAN CE device, they consist of the VLAN header, the Ethernet MAC header, and the payload. At the Ethernet PE device, the VLAN header is removed at the VLAN/QinQ subinterface's AC, and MPLS labels are inserted. The frames are then routed over the pseudowire to the HDLC PE device, where the MPLS labels are removed. The HDLC header is added before the Ethernet MAC header. The HDLC frame is then forwarded to the HDLC CE device.

If the AC is Ethernet, packets arriving from the Ethernet CE device consist of the Ethernet MAC header and the payload. At the Ethernet PE device, MPLS labels are inserted at the VLAN/QinQ subinterface's AC. The frames are then routed over the pseudowire to the HDLC PE device, where the MPLS labels are removed. The HDLC header is added before the Ethernet MAC header. The HDLC frame is then forwarded to the HDLC CE device.

The figure below shows the bridged interworking mode of HDLC-to-Ethernet interworking, with a VLAN AC on the Ethernet side.

Figure 19: HDLC-to-Ethernet — Ethernet or Bridged Interworking



HDLC-to-Ethernet — IP or Routed Interworking

To perform routed interworking, both the HDLC PE device and Ethernet PE device must be configured. The IP encapsulation over the pseudowire is performed on HDLC packets that arrive from the HDLC CE device. The address resolution is done at the HDLC PE device.

When packets arrive from the HDLC CE device, they consist of the HDLC header, the IPv4 header, and the payload. At the HDLC PE device, the HDLC header is removed, and MPLS labels are inserted. The frames are then routed over the pseudowire to the Ethernet PE device, where the MPLS labels are removed. On the Ethernet side, there are two possibilities. The attachment circuit (AC) is either Ethernet or VLAN.

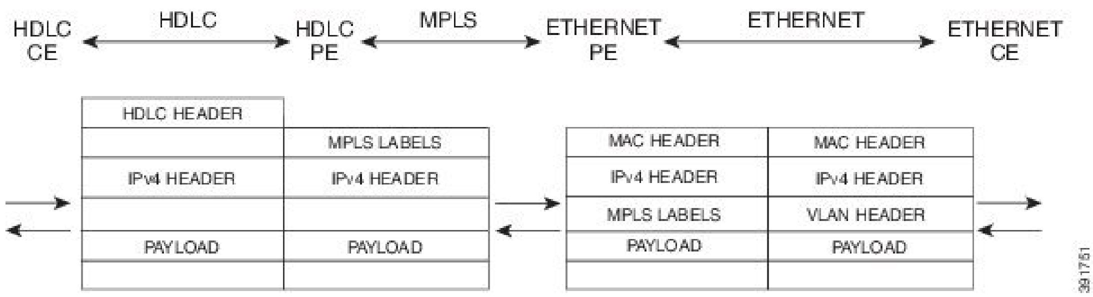
For an Ethernet attachment circuit (AC), the packets are forwarded to the Ethernet CE device, as is. For a VLAN AC, VLAN headers are added at the VLAN/QinQ subinterface’s AC. The Ethernet VLAN frame is then forwarded to the VLAN CE device.

In the opposite direction (Ethernet / VLAN to HDLC), the VLAN header is present in the incoming packet, if the AC is VLAN. So, when packets arrive from the VLAN CE device, they consist of the VLAN header, the Ethernet MAC header, and the payload. At the Ethernet PE device, the MAC header is removed, the VLAN header is removed at the VLAN/QinQ subinterface's AC, and MPLS labels are inserted. The frames are then routed over the pseudowire to the HDLC PE device, where the MPLS labels are removed. The HDLC header is added before the IPv4 header. The HDLC frame is then forwarded to the HDLC CE device.

If the AC is Ethernet, packets arriving from the Ethernet CE device consist of the Ethernet MAC header and the payload. At the Ethernet PE device, the MAC header is removed, and MPLS labels are inserted. The frames are then routed over the pseudowire to the HDLC PE device, where the MPLS labels are removed. The HDLC header is added before the IPv4 header. The HDLC frame is then forwarded to the HDLC CE device.

The figure below shows the routed interworking mode of HDLC-to-Ethernet interworking, with a VLAN AC on the Ethernet side.

Figure 20: HDLC-to-Ethernet — IP or Routed interworking



ATM Local Switching

- ATM like-to-like local switching allows switching data between two physical interfaces where both the segments are of ATM type. The two interfaces must be on the same PE router. The table below lists the supported ATM local switching combinations.

Table 12: ATM local switching - supported combinations

	Same port Point-to-Point	Different port Point-to-Point	Same Port Multipoint	Different Port Multipoint
Port Mode	No	No	No	No
VC-to-VC AAL0	Yes	Yes	Yes	Yes
VC-to-VC AAL5	Yes	Yes	Yes	Yes
VP-to-VP AAL0	No	No	Yes	Yes
VP-to-VP AAL5	No	No	No	No

VC-to-VC Local Switching

VC-to-VC local switching transports cells between two ATM attachment VCs on the same or different port on the PE router. The cells coming to the PE router can be AAL0 or AAL5 encapsulated ATM packets. ATM VC-to-VC local switching can be configured either on point-to-point interface or on multipoint interface.

There are two operation modes for managing OAM cells over ATM local switching interfaces:

- OAM transparent mode: In this mode, the PE router transports F5 OAM cells transparently across local switching interfaces.
- OAM local emulation mode: In this mode, the PE router does not transport OAM cells across local switching interfaces. Instead, the interfaces locally terminate and process F5 OAM cells.

In ATM single cell relay AAL0, the ATM virtual path identifier/virtual channel identifier (VPI/VCI) values of the ingress and egress ATM interfaces of a router must match. If L2 local switching is desired between two ATM VPIs and VCIs, which are on two different interfaces and have values that do not match, ATM AAL5 should be selected. However, if ATM AAL5 uses OAM transparent mode, the VPI and VCI values must match.

ATM OAM can be configured on ATM VC mode local switching AC using the **oam-ac emulation-enable**and **oam-pvc manage** commands. When emulation is enabled on the AC, all OAM cells going through the AC are punted to RP for local processing. The ATM common component processes OAM cells and forwards the cells towards the local CE router. This helps to detect the failures on the PE router by monitoring the response at the CE router end. When the **oam-pvc manage** command is enabled on the AC, the PVC generates end-to-end OAM loopback cells that verify connectivity on the VC.

The following example shows a sample configuration on the ATM PE router:

```
configure terminal
interface atm 4/0.50 multipoint
 no ip address
  no atm enable-ilmi-trap
pvc 100/100 l2transport
 encapsulation aal5
 oam-ac emulation-enable
 oam-pvc manage
interface atm 5/0.100 multipoint
 no ip address
 no atm enable-ilmi-trap
 pvc 100/100 l2transport
  encapsulation aal5
  oam-ac emulation-enable
  oam-pvc manage
connect atm_ls atm 4/0 100/100 atm 5/0 100/100
```

VP-to-VP Local Switching

VP-to-VP local switching transports cells between two VPs on the same port or different ports on the PE router. The cells coming to the PE router can be AAL0 encapsulated ATM packets only. ATM VP-to-VP local switching can be configured only on multipoint interfaces.

There are two operation modes for managing OAM cells over ATM local switching interfaces:

- OAM transparent mode: In this mode, the PE router transports F4 OAM cells transparently across local switching interfaces.
- OAM local emulation mode: In this mode, the PE router do not transport OAM cells across local switching interfaces. Instead, the interfaces locally terminate and process F4 OAM cells.

In ATM single cell relay AAL0, the ATM VPI values of the ingress and egress ATM interfaces on a router must match. If L2 switching is desired between two ATM VPIs which are on two different interfaces and have values that do not match, ATM AAL5 should be selected. If ATM AAL5 uses OAM transparent mode, the VPI value must match. Currently, the ATM VP-to-VP local switching supports only AAL0 encapsulation.

The following example shows a sample configuration on the ATM PE router:

```
configure terminal
interface atm 4/0.100 multipoint
 no ip address
 no atm enable-ilmi-trap
atm pvp 100 l2transport
interface atm 5/0.100 multipoint
 no ip address
 no atm enable-ilmi-trap
atm pvp 100 l2transport
connect atm_ls atm 4/0 100 atm 5/0 100
```

PPP-to-Ethernet AToM-Routed Interworking

In this interworking type, one of the ACs is Ethernet and the other is PPP. Each link is terminated locally on the corresponding PE routers and the extracted layer 3 (L3) packets are transported over a pseudowire.

The PE routers connected to Ethernet and PPP ACs terminate their respective L2 protocols. The PPP session is terminated for both the LCP and the Network Control Protocol (NCP) layers. On the ingress PE router, after extracting L3 packets, each PE router forwards the packets over the already established pseudowire using MPoMPLS encapsulation. On the egress PE router, after performing label disposition, the packets are encapsulated based on the corresponding link layer and are sent to the respective CE router. This interworking scenario requires the support of MPoMPLS encapsulation by the PE routers.

In PPP-to-Ethernet AToM routed interworking mode IPCP is supported. Proxy IPCP is automatically enabled on the PE router when IP interworking is configured on the pseudowire. By default, the PE router gets the IP address it needs to use from the CE router. The PE router accomplishes this by sending an IPCP confreq with the IP address 0.0.0.0. The local CE router has the remote CE router's IP address configured on it. The following example shows a sample configuration on the PPP CE router:

```
interface serial2/0
 ip address 168.65.32.13 255.255.255.0
 encapsulation ppp
peer default ip address 168.65.32.14  *
```

If the remote CE router's IP address cannot be configured on the local CE router, then the remote CE router's IP address can be configured on the PE router using the **ppp ipcp address proxy** *ip address* command on the xconnect PPP interface of PE router. The following example shows a sample configuration on the PPP PE router:

```
pseudowire-class mp
 encapsulation mpls
 protocol ldp
 interworking ip
!
int se2/0
 encap ppp
 xconnect 10.0.0.2 200 pw-class mp
 ppp ipcp address proxy 168.65.32.14
```

PPP-to-Ethernet AToM-Routed Interworking using the commands associated with the L2VPN Protocol-Based CLIs feature

In this interworking type, one of the ACs is Ethernet and the other is PPP. Each link is terminated locally on the corresponding PE routers and the extracted layer 3 (L3) packets are transported over a pseudowire.

The PE routers connected to Ethernet and PPP ACs terminate their respective L2 protocols. The PPP session is terminated for both the LCP and the Network Control Protocol (NCP) layers. On the ingress PE router, after extracting L3 packets, each PE router forwards the packets over the already established pseudowire using MPoMPLS encapsulation. On the egress PE router, after performing label disposition, the packets are encapsulated based on the corresponding link layer and are sent to the respective CE router. This interworking scenario requires the support of MPoMPLS encapsulation by the PE routers.

In PPP-to-Ethernet AToM routed interworking mode IPCP is supported. Proxy IPCP is automatically enabled on the PE router when IP interworking is configured on the pseudowire. By default, the PE router gets the IP address it needs to use from the CE router. The PE router accomplishes this by sending an IPCP confreq with the IP address 0.0.0.0. The local CE router has the remote CE router's IP address configured on it. The following example shows a sample configuration on the PPP CE router:

```
interface serial2/0
 ip address 168.65.32.13 255.255.255.0
 encapsulation ppp
peer default ip address 168.65.32.14  *
```

If the remote CE router's IP address cannot be configured on the local CE router, then the remote CE router's IP address can be configured on the PE router using the **ppp ipcp address proxy** *ip address* command on the xconnect PPP interface of PE router. The following example shows a sample configuration on the PPP PE router:

```
template type pseudowire mp
 encapsulation mpls
 protocol ldp
 interworking ip
!
int se2/0
 encap ppp
interface pseudowire 100
 source template type pseudowire mp
 neighbor 33.33.33.33 1
!
l2vpn xconnect context con1
 ppp ipcp address proxy 168.65.32.14
```

Static IP Addresses for L2VPN Interworking for PPP

If the PE router needs to perform address resolution with the local CE router for PPP, configure the remote CE router's IP address on the PE router. Use the **ppp ipcp address proxy** command with the remote CE router's IP address on the PE router's xconnect PPP interface. The following example shows a sample configuration:

```
pseudowire-class ip-interworking
 encapsulation mpls
 interworking ip
interface Serial2/0
 encapsulation ppp
 xconnect 10.0.0.2 200 pw-class ip-interworking
 ppp ipcp address proxy 10.65.32.14
```

You can also configure the remote CE router's IP address on the local CE router with the **peer default ip address** command if the local CE router performs address resolution.

Static IP Addresses for L2VPN Interworking for PPP using the commands associated with the L2VPN Protocol-Based CLIs feature

If the PE router needs to perform address resolution with the local CE router for PPP, configure the remote CE router's IP address on the PE router. Use the **ppp ipcp address proxy** command with the remote CE router's IP address on the PE router's xconnect PPP interface. The following example shows a sample configuration:

```
template type pseudowire ip-interworking
```

```
 encapsulation mpls
 interworking ip
interface Serial2/0
 encapsulation ppp
interface pseudowire 100
 source template type pseudowire ip-interworking
 neighbor 10.0.0.2 200
!
l2vpn xconnect context con1
 ppp ipcp address proxy 10.65.32.14
```

You can also configure the remote CE router's IP address on the local CE router with the **peer default ip address** command if the local CE router performs address resolution.

How to Configure L2VPN Interworking

Configuring L2VPN Interworking

L2VPN interworking allows you to connect disparate ACs. Configuring L2VPN interworking feature requires that you add the **interworking** command to the list of commands that make up the pseudowire. The steps for configuring the pseudowire for L2VPN interworking are included in this section. You use the **interworking**command as part of the overall AToM configuration. For specific instructions on configuring AToM, see the Any Transport over MPLS document.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *name*
4. **encapsulation** {**mpls** | **l2tpv3**}
5. **interworking** {**ethernet** | **ip**}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *name* **Example:**	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.

	Command or Action	Purpose
	`Router(config)# pseudowire-class class1`	
Step 4	**encapsulation** {**mpls** | **l2tpv3**} **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation, which is either **mpls** or **l2tpv3**.
Step 5	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 6	**end** **Example:** `Router(config-pw)# end`	Exits pseudowire class configuration mode and returns to privileged EXEC mode.

Verifying the L2VPN Configuration

You can verify L2VPN configuration using the following steps:

- You can issue the **show arp** command between the CE routers to ensure that data is being sent:

```
Router# show arp
Protocol   Address        Age (min)   Hardware Addr    Type    Interface
Internet   10.1.1.5             134   0005.0032.0854   ARPA    FastEthernet0/0/0
Internet   10.1.1.7               -   0005.0032.0000   ARPA    FastEthernet0/0/0
```

- You can issue the **ping** command between the CE routers to ensure that data is being sent:

```
Router# ping 10.1.1.5
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.1.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

- You can verify the AToM configuration by using the **show mpls l2transport vc detail** command.

Configuring L2VPN Interworking using the commands associated with the L2VPN Protocol-Based CLIs feature

L2VPN Interworking allows you to connect disparate attachment circuits. Configuring the L2VPN Interworking feature requires that you add the **interworking** command to the list of commands that make up the pseudowire. The steps for configuring the pseudowire for L2VPN Interworking are included in this section. You use the **interworking**command as part of the overall AToM or L2TPv3 configuration. For specific instructions on configuring AToM or L2TPv3, see the following documents:

- Layer 2 Tunnel Protocol Version 3

- Any Transport over MPLS

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hw-module slot** *slot-number* **np mode feature**
4. **interface pseudowire** *number*
5. **encapsulation** {**mpls** | **l2tpv3**}
6. **interworking** {**ethernet** | **ip**}
7. **neighbor** *peer-address vcid-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**hw-module slot** *slot-number* **np mode feature** **Example:** `Router(config)# hw-module slot 3 np mode feature`	(Optional) Enables L2VPN Interworking functionality on the Cisco 12000 series router. **Note** Enter this command only on a Cisco 12000 series Internet router if you use L2TPv3 for L2VPN Interworking on an ISE (Engine 3) or Engine 5 interface. In this case, you must first enable the L2VPN feature bundle on the line card by entering the **hw-module slot** *slot-number* **np mode feature** command.
Step 4	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 1`	Establishes an interface pseudowire with a value that you specify and enters pseudowire class configuration mode.
Step 5	**encapsulation** {**mpls** | **l2tpv3**} **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation, which is either **mpls** or **l2tpv3**.
Step 6	**interworking** {**ethernet** | **ip**} **Example:**	Specifies the type of pseudowire and the type of traffic that can flow across it.

	Command or Action	Purpose
	`Router(config-pw)# interworking ip`	**Note** On the Cisco 12000 series Internet router, Ethernet (bridged) interworking is not supported for L2TPv3. After you configure the L2TPv3 tunnel encapsulation for the pseudowire using the **encapsulation l2tpv3**command, you cannot enter the **interworking ethernet** command.
Step 7	**neighbor** *peer-address vcid-value* **Example:** `Router(config-pw)#  neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.

Verifying the L2VPN Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

You can verify L2VPN configuration using the following commands:

- You can issue the **show arp** command between the CE routers to ensure that data is being sent:

```
Device# show arp
Protocol   Address        Age (min)   Hardware Addr    Type    Interface
Internet   10.1.1.5           134    0005.0032.0854   ARPA    FastEthernet0/0/0
Internet   10.1.1.7             -    0005.0032.0000   ARPA    FastEthernet0/0/0
```

- You can issue the **ping** command between the CE routers to ensure that data is being sent:

```
Device# ping 10.1.1.5
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.1.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

- You can verify the AToM configuration by using the **show l2vpn atom vc detail** command.

Configuring Ethernet VLAN-to-ATM AAL5 Interworking

This section explains the following AToM configurations:

ATM AAL5-to-Ethernet Port

You can configure the ATM AAL5-to-Ethernet Port feature on a PE1 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*

6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface atm** *slot* / *subslot* / *port* **.** *subinterface number*
10. **pvc** [*name*] *vpi* / *vci* **12transport**
11. **encapsulation aal5snap**
12. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.

	Command or Action	Purpose
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface atm** *slot* / *subslot* / *port* **.** *subinterface number* **Example:** `Router(config-pw)# interface atm 2/0/0.1`	Configures an ATM interface and enters interface configuration mode.
Step 10	**pvc** [*name*] *vpi* / *vci* **12transport** **Example:** `Router(config-subif)# pvc 0/200 l2transport`	Assigns a name to an ATM permanent virtual circuit (PVC) and enters ATM virtual circuit configuration mode.
Step 11	**encapsulation aal5snap** **Example:** `Router(config-if-atm-member)# encapsulation aal5snap`	Configures the ATM AAL and encapsulation type for an ATM VC.
Step 12	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:** `Router(config-if-atm-member)# xconnect 10.0.0.200 140 pw-class atm-eth`	Binds an AC to a pseudowire and configures an AToM static pseudowire.
Step 13	**end** **Example:** `Router(config-if-xconn)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

ATM AAL5-to-Ethernet Port using the commands associated with the L2VPN Protocol-Based CLIs feature

You can configure the ATM AAL5-to-Ethernet Port feature on a PE1 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **template type pseudowire** [*pw-class-name*]
7. **encapsulation mpls**

8. **interworking** {**ethernet** | **ip**}
9. **interface atm** *slot* / *subslot* / *port* **.** *subinterface number*
10. **pvc** [*name*] *vpi* / *vci* **12transport**
11. **encapsulation aal5snap**
12. **end**
13. **interface pseudowire** *number*
14. **source template type pseudowire** *template-name*
15. **neighbor** *peer-address vcid-value*
16. **exit**
17. **exit**
18. **l2vpn xconnect context** *context-name*
19. **member pseudowire** *interface-number*
20. **member** *ip-address vc-id* **encapsulation mpls**
21. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**template type pseudowire** [*pw-class-name*] **Example:**	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.

	Command or Action	Purpose
	`Router(config-if)# template type pseudowire atm-eth`	
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface atm** *slot* / *subslot* / *port* **.** *subinterface number* **Example:** `Router(config-pw)# interface atm 2/0/0.1`	Configures an ATM interface and enters interface configuration mode.
Step 10	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-subif)# pvc 0/200 l2transport`	Assigns a name to an ATM permanent virtual circuit (PVC) and enters ATM virtual circuit configuration mode.
Step 11	**encapsulation aal5snap** **Example:** `Router(config-if-atm-member)# encapsulation aal5snap`	Configures the ATM AAL and encapsulation type for an ATM VC.
Step 12	**end** **Example:** `Router(config-if-atm-member)# end`	Exits to privileged EXEC mode.
Step 13	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 14	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire atm-eth`	Configures the source template of type pseudowire named atm-eth.
Step 15	**neighbor** *peer-address vcid-value* **Example:**	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.

	Command or Action	Purpose
	`Router(config-if)#  neighbor 10.0.0.200 140`	
Step 16	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 17	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 18	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 19	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 20	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.200 140 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 21	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

ATM AAL5-to-Ethernet Port on a PE2 Router

You can configure the ATM AAL5-to-Ethernet Port feature on a PE2 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}

9. **interface** *type* *slot* / *subslot* / *port*
10. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.

	Command or Action	Purpose
Step 9	**interface** *type* *slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface gigabitethernet 5/1/0`	Configure an interface and enters interface configuration mode.
Step 10	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:** `Router(config-if)# xconnect 10.0.0.100 140 pw-class atm-eth`	Binds an AC to a pseudowire and configures an AToM static pseudowire.
Step 11	**end** **Example:** `Router(config-if-xconn)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

What to do next

Note When configuring bridged interworking, the PE2 router configuration does not include the **interworking ethernet** command because it is treated as like-to-like, and also because the AC is already an Ethernet port. However, when configuring routed interworking, the **interworking ip** command is required.

ATM AAL5-to-Ethernet Port on a PE2 Router using the commands associated with the L2VPN Protocol-Based CLIs feature

You can configure the ATM AAL5-to-Ethernet Port feature on a PE2 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **template type pseudowire** [*pseudowire-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface** *type slot* / *subslot* / *port*
10. **end**
11. **interface pseudowire** *number*
12. **source template type pseudowire** *template-name*
13. **neighbor** *peer-address vcid-value*
14. **exit**
15. **l2vpn xconnect context** *context-name*

16. **member pseudowire** *interface-number*
17. **member** *ip-address vc-id* **encapsulation mpls**
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**template type pseudowire** [*pseudowire-name*] **Example:** `Router(config)# template type pseudowire atm-eth`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.

	Command or Action	Purpose
Step 9	**interface** *type* *slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface gigabitethernet 5/1/0`	Configure an interface and enters interface configuration mode.
Step 10	**end** **Example:** `Router(config-pw)# end`	Exits to privileged EXEC mode.
Step 11	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 12	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire atm-eth`	Configures the source template of type pseudowire named atm-eth
Step 13	**neighbor** *peer-address* *vcid-value* **Example:** `Router(config-if)#  neighbor 10.0.0.100 140`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 14	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 15	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 16	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 17	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.100 140 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.

	Command or Action	Purpose
Step 18	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

What to do next

Note When configuring bridged interworking, the PE2 router configuration does not include the **interworking ethernet** command because it is treated as like-to-like, and also because the AC is already an Ethernet port. However, when configuring routed interworking, the **interworking ip** command is required.

ATM AAL5-to-Ethernet VLAN 802.1Q on a PE1 Router

You can configure the ATM AAL5-to-Ethernet VLAN 802.1Q feature on a PE1 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface atm** *slot* / *subslot* / *port* **.** *subinterface number*
10. **pvc** [*name*] *vpi* / *vci* **12transport**
11. **encapsulation aal5snap**
12. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface atm** *slot* / *subslot* / *port* **.** *subinterface number* **Example:** `Router(config-pw)# interface atm 2/0/0.1`	Configure an ATM interface and enters interface configuration mode.
Step 10	**pvc** [*name*] *vpi* / *vci* **12transport** **Example:** `Router(config-subif)# pvc 0/200 l2transport`	Assigns a name to an ATM permanent virtual circuit (PVC) and enters ATM virtual circuit configuration mode.
Step 11	**encapsulation aal5snap** **Example:**	Configures the ATM AAL and encapsulation type for an ATM VC.

	Command or Action	Purpose
	`Router(config-if-atm-member)# encapsulation aal5snap`	
Step 12	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:** `Router(config-if-atm-member)# xconnect 10.0.0.200 140 pw-class atm-eth`	Binds an AC to a pseudowire and configures an AToM static pseudowire.
Step 13	**end** **Example:** `Router(config-if-xconn)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

ATM AAL5-to-Ethernet VLAN 802.1Q on a PE1 Router using the commands associated with the L2VPN Protocol-Based CLIs feature

You can configure the ATM AAL5-to-Ethernet VLAN 802.1Q feature on a PE1 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **template type pseudowire** [*pseudowire-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface atm** *slot* / *subslot* / *port* **.** *subinterface number*
10. **pvc** [*name*] *vpi* / *vci* **12transport**
11. **encapsulation aal5snap**
12. **end**
13. **interface pseudowire** *number*
14. **source template type pseudowire** *template-name*
15. **neighbor** *peer-address vcid-value*
16. **exit**
17. **l2vpn xconnect context** *context-name*
18. **member pseudowire** *interface-number*
19. **member** *ip-address vc-id* **encapsulation mpls**
20. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**template type pseudowire** [*pseudowire-name*] **Example:** `Router(config)# template type pseudowire atm-eth`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface atm** *slot* / *subslot* / *port* . *subinterface number* **Example:** `Router(config-pw)# interface atm 2/0/0.1`	Configure an ATM interface and enters interface configuration mode.

	Command or Action	Purpose
Step 10	**pvc** [*name*] *vpi* / *vci* **l2transport** **Example:** `Router(config-subif)# pvc 0/200 l2transport`	Assigns a name to an ATM permanent virtual circuit (PVC) and enters ATM virtual circuit configuration mode.
Step 11	**encapsulation aal5snap** **Example:** `Router(config-if-atm-member)# encapsulation aal5snap`	Configures the ATM AAL and encapsulation type for an ATM VC.
Step 12	**end** **Example:** `Router(config-if-atm-member)# end`	Exits to privileged EXEC mode.
Step 13	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 14	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire atm-eth`	Configures the source template of type pseudowire named atm-eth
Step 15	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)#  neighbor 10.0.0.200 140`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 16	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 17	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 18	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.

	Command or Action	Purpose
Step 19	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.200 140 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 20	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

ATM AAL5-to-Ethernet VLAN 802.1Q on a PE2 router

You can configure the ATM AAL5-to-Ethernet VLAN 802.1Q feature on a PE2 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface** *type slot* / *subslot* / *port* **.** *subinterface-number*
10. **encapsulation dot1q** *vlan-id*
11. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:**	Establishes the label distribution protocol for the platform.

	Command or Action	Purpose
	`Router(config)# mpls label protocol ldp`	
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface** *type slot* / *subslot* / *port* . *subinterface-number* **Example:** `Router(config-pw)# interface gigabitethernet 5/1/0.3`	Configures an interface and enters interface configuration mode.
Step 10	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if)# encapsulation dot1q 1525`	Enables IEEE 802.1Q encapsulation of traffic on a specified sub interface in a VLAN.
Step 11	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:** `Router(config-if)# xconnect 10.0.0.100 140 pw-class atm-eth`	Binds an AC to a pseudowire and configures an AToM static pseudowire.
Step 12	**end** **Example:**	Exits xconnect configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Router(config-if-xconn)# end`	

What to do next

> **Note** In the case of ATM AAl5-to-VLAN, the PE2 router configuration includes the **interworking**command for both bridged and routed interworking.

> **Note** To verify the L2VPN interworking status and check the statistics, refer to the Verifying L2VPN Interworking, on page 249.

ATM AAL5-to-Ethernet VLAN 802.1Q on a PE2 router using the commands associated with the L2VPN Protocol-Based CLIs feature

You can configure the ATM AAL5-to-Ethernet VLAN 802.1Q feature on a PE2 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **template type pseudowire** [*pseudowire-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface** *type slot* / *subslot* / *port* **.** *subinterface-number*
10. **encapsulation dot1q** *vlan-id*
11. **end**
12. **interface pseudowire** *number*
13. **source template type pseudowire** *template-name*
14. **neighbor** *peer-address vcid-value*
15. **exit**
16. **l2vpn xconnect context** *context-name*
17. **member pseudowire** *interface-number*
18. **member** *ip-address vc-id* **encapsulation mpls**
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configure an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**template type pseudowire** [*pseudowire-name*] **Example:** `Router(config)# template type pseudowire atm-eth`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface** *type slot* / *subslot* / *port* . *subinterface-number* **Example:**	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
	`Router(config-pw)# interface gigabitethernet 5/1/0.3`	
Step 10	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if)# encapsulation dot1q 1525`	Enables IEEE 802.1Q encapsulation of traffic on a specified sub interface in a VLAN.
Step 11	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 12	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 13	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire atm-eth`	Configures the source template of type pseudowire named atm-eth
Step 14	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)#  neighbor 10.0.0.100 140`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 15	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 16	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 17	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 18	**member** *ip-address vc-id* **encapsulation mpls** **Example:**	Creates the VC to transport the Layer 2 packets.

	Command or Action	Purpose
	`Router(config-xconnect)# member 10.0.0.100 140 encapsulation mpls`	
Step 19	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

What to do next

Note In the case of ATM AAl5-to-VLAN, the PE2 router configuration includes the **interworking**command for both bridged and routed interworking.

Note To verify the L2VPN interworking status and check the statistics, refer to the Verifying L2VPN Interworking, on page 249.

Configuring Ethernet VLAN-to-Frame Relay Interworking

This section explains the following AToM configurations and provides examples. The Network Topology for FR-to-Ethernet AToM Bridged Interworking figure above illustrates different AToM configurations.

Frame Relay DLCI-to-Ethernet Port on a PE1 Router

You can configure the Frame Relay DLCI-to-Ethernet Port feature on a PE1 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking ethernet**
9. **interface** *type slot / subslot / port*
10. **encapsulation frame-relay**
11. **connect** *connection-name interface dlci* {*interface dlci* | **l2transport**}
12. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class fr-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking ethernet** **Example:** `Router(config-pw)# interworking ethernet`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface** *type slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface serial 2/0/0`	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 10	**encapsulation frame-relay** **Example:** `Router(config-if)# encapsulation frame-relay`	Enables Frame Relay encapsulation.
Step 11	**connect** *connection-name interface dlci* {*interface dlci* | **l2transport**} **Example:** `Router(config-if)# connect fr-vlan-1 POS2/3/1 151 l2transport`	Defines the connection between Frame Relay PVCs.
Step 12	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:** `Router(config-if)# xconnect 10.0.0.200 151 pw-class pw-class-bridge`	Binds an AC to a pseudowire and configures an AToM static pseudowire.
Step 13	**end** **Example:** `Router(config-if-xconn)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

Frame Relay DLCI-to-Ethernet Port on a PE1 Router using the commands associated with the L2VPN Protocol-Based CLIs feature

You can configure the Frame Relay DLCI-to-Ethernet Port feature on a PE1 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **template type pseudowire** [*pseudowire-name*]
7. **encapsulation mpls**
8. **interworking ethernet**
9. **interface** *type slot* / *subslot* / *port*
10. **encapsulation frame-relay**
11. **connect** *connection-name interface dlci* {*interface dlci* | **l2transport**}
12. **end**
13. **interface pseudowire** *number*
14. **source template type pseudowire** *template-name*
15. **neighbor** *peer-address vcid-value*
16. **exit**

17. **l2vpn xconnect context** *context-name*
18. **member pseudowire** *interface-number*
19. **member** *ip-address vc-id* **encapsulation mpls**
20. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**template type pseudowire** [*pseudowire-name*] **Example:** `Router(config)# template type pseudowire fr-eth`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking ethernet** **Example:** `Router(config-pw)# interworking ethernet`	Specifies the type of pseudowire and the type of traffic that can flow across it.

	Command or Action	Purpose
Step 9	**interface** *type* *slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface serial 2/0/0`	Configures an interface and enters interface configuration mode.
Step 10	**encapsulation frame-relay** **Example:** `Router(config-if)# encapsulation frame-relay`	Enables Frame Relay encapsulation.
Step 11	**connect** *connection-name interface dlci* {*interface dlci* | **l2transport**} **Example:** `Router(config-if)# connect fr-vlan-1 POS2/3/1 151 l2transport`	Defines the connection between Frame Relay PVCs.
Step 12	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 13	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 14	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire pwclass-bridge`	Configures the source template of type pseudowire named pwclass-bridge.
Step 15	**neighbor** *peer-address* *vcid-value* **Example:** `Router(config-if)#  neighbor 10.0.0.200 151`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 16	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 17	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.

	Command or Action	Purpose
Step 18	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 19	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.200 151 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 20	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

Frame Relay DLCI-to-Ethernet Port on a PE2 router

You can configure the Frame Relay DLCI-to-Ethernet Port feature on a PE2 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking ethernet**
9. **interface** *type slot* / *subslot* / *port*
10. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking ethernet** **Example:** `Router(config-pw)# interworking ethernet`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface** *type slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface gigabitethernet 2/0/0`	Configures an interface and enters interface configuration mode.
Step 10	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:** `Router(config-if)# xconnect 10.0.0.200 140 pw-class atm-eth`	Binds an AC to a pseudowire and configures an AToM static pseudowire.
Step 11	**end** **Example:** `Router(config-if-xconn)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

What to do next

> **Note**
> When configuring bridged interworking, the PE2 router configuration does not include the **interworking ethernet**command because it is treated as like-to-like, and also because the AC is already an Ethernet port. However, when configuring routed interworking, the PE2 router configuration does include the **interworking ip** command.

Frame Relay DLCI-to-Ethernet Port on a PE2 router using the commands associated with the L2VPN Protocol-Based CLIs feature

You can configure the Frame Relay DLCI-to-Ethernet Port feature on a PE2 router using the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **template type pseudowire** [*pseudowire-name*]
7. **encapsulation mpls**
8. **interworking ethernet**
9. **interface** *type slot* / *subslot* / *port*
10. **end**
11. **interface pseudowire** *number*
12. **source template type pseudowire** *template-name*
13. **neighbor** *peer-address vcid-value*
14. **exit**
15. **l2vpn xconnect context** *context-name*
16. **member pseudowire** *interface-number*
17. **member** *ip-address vc-id* **encapsulation mpls**
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**template type pseudowire** [*pseudowire-name*] **Example:** `Router(config)# template type pseudowire atm-eth`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking ethernet** **Example:** `Router(config-pw)# interworking ethernet`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface** *type slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface gigabitethernet 2/0/0`	Configures an interface and enters interface configuration mode.
Step 10	**end** **Example:** `Router(config-pw)# end`	Exits to privileged EXEC mode.
Step 11	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.

	Command or Action	Purpose
Step 12	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire atm-eth`	Configures the source template of type pseudowire named atm-eth
Step 13	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.200 140`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 14	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 15	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 16	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 17	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.200 140 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 18	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

What to do next

Note When configuring bridged interworking, the PE2 router configuration does not include the **interworking ethernet**command because it is treated as like-to-like, and also because the AC is already an Ethernet port. However, when configuring routed interworking, the PE2 router configuration does include the **interworking ip** command.

Frame Relay DLCI-to-Ethernet VLAN 802.1Q on a PE1 Router

To configure the Frame Relay DLCI-to-Ethernet VLAN 802.1Q feature on a PE1 router, use the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **frame-relay switching**
10. **interface** *type slot* / *subslot* / *port*
11. **encapsulation frame-relay**
12. **frame-relay intf-type** [**dce**]
13. **connect** *connection-name interface dlci* {*interface dlci* | **l2transport**}
14. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:**	Sets the primary or secondary IP address for an interface.

	Command or Action	Purpose
	`Router(config-if)# ip address 10.0.0.100 255.255.255.255`	
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**frame-relay switching** **Example:** `Router(config-pw)# frame-relay switching`	Enables PVC switching on a Frame Relay DCE device.
Step 10	**interface** *type slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface serial 2/0/0`	Configures an interface and enters interface configuration mode.
Step 11	**encapsulation frame-relay** **Example:** `Router(config-if)# encapsulation frame-relay`	Enables Frame Relay encapsulation.
Step 12	**frame-relay intf-type** [**dce**] **Example:** `Router(config-if)# frame-relay intf-type dce`	Configures a Frame Relay switch type.
Step 13	**connect** *connection-name interface dlci* {*interface dlci* | **l2transport**} **Example:** `Router(config-if)# connect one serial0 16 serial1 100`	Defines the connection between Frame Relay PVCs.
Step 14	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:**	Binds an AC to a pseudowire and configures an AToM static pseudowire.

	Command or Action	Purpose
	`Router(config-if)# xconnect 10.0.0.200 140 pw-class atm-eth`	
Step 15	**end** **Example:** `Router(config-if-xconn)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

Frame Relay DLCI-to-Ethernet VLAN 802.1Q on a PE1 Router using the commands associated with the L2VPN Protocol-Based CLIs feature

To configure the Frame Relay DLCI-to-Ethernet VLAN 802.1Q feature on a PE1 router, use the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **template type pseudowire** [*pseudowire-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **frame-relay switching**
10. **interface** *type slot* / *subslot* / *port*
11. **encapsulation frame-relay**
12. **frame-relay intf-type** [**dce**]
13. **connect** *connection-name interface dlci* {*interface dlci* | **l2transport**}
14. **end**
15. **interface pseudowire** *number*
16. **source template type pseudowire** *template-name*
17. **neighbor** *peer-address vcid-value*
18. **exit**
19. **l2vpn xconnect context** *context-name*
20. **member pseudowire** *interface-number*
21. **member** *ip-address vc-id* **encapsulation mpls**
22. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**template type pseudowire** [*pseudowire-name*] **Example:** `Router(config)# template type pseudowire atm-eth`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**frame-relay switching** **Example:** `Router(config-pw)# frame-relay switching`	Enables PVC switching on a Frame Relay DCE device.
Step 10	**interface** *type slot* / *subslot* / *port* **Example:** `Router(config-pw)# interface serial 2/0/0`	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 11	**encapsulation frame-relay** **Example:** `Router(config-if)# encapsulation frame-relay`	Enables Frame Relay encapsulation.
Step 12	**frame-relay intf-type** [**dce**] **Example:** `Router(config-if)# frame-relay intf-type dce`	Configures a Frame Relay switch type.
Step 13	**connect** *connection-name interface dlci* {*interface dlci* | **l2transport**} **Example:** `Router(config-if)# connect one serial0 16 serial1 100`	Defines the connection between Frame Relay PVCs.
Step 14	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 15	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 16	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire atm-eth`	Configures the source template of type pseudowire named atm-eth
Step 17	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)#  neighbor 10.0.0.200 140`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 18	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 19	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.

	Command or Action	Purpose
Step 20	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 21	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.200 140 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 22	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

Frame Relay DLCI-to-Ethernet VLAN 802.1Q on a PE2 Router

To configure the Frame Relay DLCI-to-Ethernet VLAN 802.1Q feature on a PE2 router, use the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface** *type slot* / *subslot* / *port* **.** *subinterface-number*
10. **encapsulation dot1q** *vlan-id*
11. **xconnect** *ip-address vc-id* **pw-class** *pw-class-name*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.
Step 9	**interface** *type slot* / *subslot* / *port* . *subinterface-number* **Example:** `Router(config-pw)# interface gigabitethernet 5/1/0.3`	Configures an interface and enters interface configuration mode.
Step 10	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if)# encapsulation dot1q 1525`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 11	**xconnect** *ip-address vc-id* **pw-class** *pw-class-name* **Example:**	Binds an AC to a pseudowire and configures an AToM static pseudowire.

	Command or Action	Purpose
	`Router(config-if)# xconnect 10.0.0.100 140 pw-class atm-eth`	
Step 12	**end** **Example:** `Router(config-if-xconn)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

What to do next

Note In the case of an Frame Relay DLCI-to-VLAN, the PE2 router configuration includes the **interworking**command for both bridged and routed interworking.

Note To verify the L2VPN interworking status and check the statistics, refer to the Verifying L2VPN Interworking, on page 249.

Frame Relay DLCI-to-Ethernet VLAN 802.1Q on a PE2 Router using the commands associated with the L2VPN Protocol-Based CLIs feature

To configure the Frame Relay DLCI-to-Ethernet VLAN 802.1Q feature on a PE2 router, use the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **pseudowire-class** [*pw-class-name*]
7. **encapsulation mpls**
8. **interworking** {**ethernet** | **ip**}
9. **interface** *type slot* / *subslot* / *port* . *subinterface-number*
10. **encapsulation dot1q** *vlan-id*
11. **end**
12. **interface pseudowire** *number*
13. **source template type pseudowire** *template-name*
14. **exit**
15. **l2vpn xconnect context** *context-name*
16. **member pseudowire** *interface-number*
17. **member** *ip-address vc-id* **encapsulation mpls**

18. **interworking ip**
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:** `Router(config)# mpls label protocol ldp`	Establishes the label distribution protocol for the platform.
Step 4	**interface** *type number* **Example:** `Router(config)# interface loopback 100`	Configures an interface type and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.0.0.100 255.255.255.255`	Sets the primary or secondary IP address for an interface.
Step 6	**pseudowire-class** [*pw-class-name*] **Example:** `Router(config-if)# pseudowire-class atm-eth`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 7	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 8	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	Specifies the type of pseudowire and the type of traffic that can flow across it.

	Command or Action	Purpose
Step 9	**interface** *type slot* / *subslot* / *port* . *subinterface-number* **Example:** `Router(config-pw)# interface gigabitethernet 5/1/0.3`	Configures an interface and enters interface configuration mode.
Step 10	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if)# encapsulation dot1q 1525`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 11	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.
Step 12	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 13	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire ether-pw`	Configures the source template of type pseudowire named ether-pw.
Step 14	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.
Step 15	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 16	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 17	**member** *ip-address vc-id* **encapsulation mpls** **Example:** `Router(config-xconnect)# member 10.0.0.100 140 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.

	Command or Action	Purpose
Step 18	**interworking ip** **Example:** `Router(config-xconnect)# interworking ip`	Establishes an L2VPN cross connect context.
Step 19	**end** **Example:** `Router(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

What to do next

> **Note** In the case of an Frame Relay DLCI-to-VLAN, the PE2 router configuration includes the **interworking**command for both bridged and routed interworking.

> **Note** To verify the L2VPN interworking status and check the statistics, refer to the Verifying L2VPN Interworking, on page 249.

Configuring HDLC-to-Ethernet Interworking

HDLC-to-Ethernet Bridged Interworking on a HDLC PE Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation mpls**
5. **interworking ethernet**
6. **interface** *type slot/subslot /port* [**.** *subinterface*]
7. **no ip address** [*ip-address mask*] [**secondary**]
8. **xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class pw-iw-ether`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**interworking ethernet** **Example:** `Device(config-pw-class)# interworking ethernet`	Specifies Ethernet as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 6	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config-pw-class)# interface serial 3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 7	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 8	**xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*] **Example:** `Device(config-if)# xconnect 198.51.100.2 123 pseudowire-class pw-iw-ether`	Creates the virtual circuit (VC) to transport the Layer 2 packets.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Bridged Interworking on a HDLC PE Device Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **template type pseudowire** *name*
4. **encapsulation mpls**
5. **exit**
6. **interface pseudowire** *number*
7. **source template type pseudowire** *name*
8. **encapsulation mpls**
9. **neighbor** *peer-address vc id-value*
10. **signaling protocol ldp**
11. **no shutdown**
12. **exit**
13. **l2vpn xconnect context** *context-name*
14. **interworking ethernet**
15. **member** *interface-type-number*
16. **member pseudowire** *interface-number*
17. **no shutdown**
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** *name* **Example:** `Device# template type pseudowire temp5`	Creates a template pseudowire with a name that you specify and enters template configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-template)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to global configuration mode.
Step 6	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 107`	Establishes an interface pseudowire with a value that you specify and enters interface configuration mode.

	Command or Action	Purpose
Step 7	**source template type pseudowire** *name* **Example:** `Device(config-if)# source template type pseudowire temp5`	Configures the source template of type pseudowire named temp5.
Step 8	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 9	**neighbor** *peer-address vc id-value* **Example:** `Device(config-if)# neighbor 10.0.0.11 107`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 10	**signaling protocol ldp** **Example:** `Device(config-if)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.
Step 11	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts the interface pseudowire.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 13	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates an L2VPN cross-connect context and enters xconnect configuration mode.
Step 14	**interworking ethernet** **Example:** `Device(config-xconnect)# interworking ethernet`	Specifies Ethernet as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 15	**member** *interface-type-number* **Example:** `Device(config-xconnect)# member serial 0/1/0:0`	Specifies the location of the member interface.
Step 16	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 107`	Specifies a member pseudowire to form an L2VPN cross connect.
Step 17	**no shutdown** **Example:** `Device(config-xconnect)# no shutdown`	Restarts the member interface.

	Command or Action	Purpose
Step 18	**end** **Example:** `Device(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Bridged Interworking (Port Mode) on an Ethernet PE Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation mpls**
5. **interworking ethernet**
6. **interface** *type slot/subslot /port* [**.** *subinterface*]
7. **encapsulation mpls**
8. **xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class pw-iw-ether`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**interworking ethernet** **Example:** `Device(config-pw-class)# interworking ethernet`	Specifies Ethernet as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.

	Command or Action	Purpose
Step 6	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config-pw-class)# interface gigabitethernet 4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.
Step 7	**encapsulation mpls** **Example:** `Device(config-subif)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 8	**xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*] **Example:** `Device(config-subif)# xconnect 198.51.100.2 123 pseudowire-class pw-iw-ether`	Creates the virtual circuit (VC) to transport the Layer 2 packets.
Step 9	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Bridged Interworking (Port Mode) on an Ethernet PE Device Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot*/*subslot* /*port* [**.** *subinterface*]
4. **encapsulation mpls**
5. **no ip address**
6. **no shutdown**
7. **exit**
8. **template type pseudowire** *name*
9. **encapsulation mpls**
10. **exit**
11. **interface pseudowire** *number*
12. **source template type pseudowire** *name*
13. **encapsulation mpls**
14. **neighbor** *peer-address vc id-value*
15. **signaling protocol ldp**
16. **no shutdown**
17. **exit**

18. **l2vpn xconnect context** *context-name*
19. **interworking ethernet**
20. **member** *interface-type-number*
21. **member pseudowire** *interface-number*
22. **no shutdown**
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config)# interface fastethernet 4/0/0.1`	Specifies the subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.
Step 4	**encapsulation mpls** **Example:** `Device(config-subif)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**no ip address** **Example:** `Device(config-subif)# no ip address`	Disables IP processing.
Step 6	**no shutdown** **Example:** `Device(config-subif)# no shutdown`	Restarts the Fast Ethernet subinterface.
Step 7	**exit** **Example:** `Device(config-subif)# exit`	Exits subinterface configuration mode and returns to global configuration mode.
Step 8	**template type pseudowire** *name* **Example:** `Device(config)# template type pseudowire temp4`	Creates a template pseudowire with a name that you specify and enters template configuration mode.

	Command or Action	Purpose
Step 9	**encapsulation mpls** **Example:** `Device(config-template)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 10	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to global configuration mode.
Step 11	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 109`	Establishes an interface pseudowire with a value that you specify and enters interface configuration mode.
Step 12	**source template type pseudowire** *name* **Example:** `Device(config-if)# source template type pseudowire temp4`	Configures the source template of type pseudowire named temp4.
Step 13	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 14	**neighbor** *peer-address vc id-value* **Example:** `Device(config-if)# neighbor 10.0.0.15 109`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 15	**signaling protocol ldp** **Example:** `Device(config-if)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.
Step 16	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts the interface pseudowire.
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 18	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con2`	Creates an L2VPN cross-connect context and enters xconnect configuration mode.
Step 19	**interworking ethernet** **Example:**	Specifies Ethernet as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.

	Command or Action	Purpose
	`Device(config-xconnect)# interworking ethernet`	
Step 20	**member** *interface-type-number* **Example:** `Device(config-xconnect)# member fastethernet 4/0/0.1`	Specifies the location of the member interface.
Step 21	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 109`	Specifies a member pseudowire to form an L2VPN cross connect.
Step 22	**no shutdown** **Example:** `Device(config-xconnect)# no shutdown`	Restarts the member interface.
Step 23	**end** **Example:** `Device(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Bridged Interworking (dot1q and QinQ Modes) on an Ethernet PE Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation mpls**
5. **interworking ethernet**
6. **interface** *type slot/subslot /port* [**.** *subinterface*]
7. **encapsulation dot1q** *vlan-id***second dot1q** *vlan-id*
8. **xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class pw-iw-ether`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**interworking ethernet** **Example:** `Device(config-pw-class)# interworking ethernet`	Specifies Ethernet as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 6	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config-pw-class)# interface gigabitethernet 4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.
Step 7	**encapsulation dot1q** *vlan-id***second dot1q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1q 100 second dot1q 200`	Defines the matching criteria to map QinQ ingress frames on an interface to the appropriate service instance.
Step 8	**xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*] **Example:** `Device(config-subif)# xconnect 198.51.100.2 123 pseudowire-class pw-iw-ether`	Creates the virtual circuit (VC) to transport the Layer 2 packets.
Step 9	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Bridged Interworking (dot1q and QinQ Modes) on an Ethernet PE Device Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot*/*subslot* /*port* [**.** *subinterface*]

4. **encapsulation dot1q** *vlan-id* **second dot1q** *vlan-id*
5. **no ip address**
6. **no shutdown**
7. **exit**
8. **template type pseudowire** *name*
9. **encapsulation mpls**
10. **exit**
11. **interface pseudowire** *number*
12. **source template type pseudowire** *name*
13. **encapsulation mpls**
14. **neighbor** *peer-address vc id-value*
15. **signaling protocol ldp**
16. **no shutdown**
17. **exit**
18. **l2vpn xconnect context** *context-name*
19. **interworking ethernet**
20. **member** *interface-type-number*
21. **member pseudowire** *interface-number*
22. **no shutdown**
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot***/***subslot* **/***port* [**.** *subinterface*] **Example:** `Device(config)# interface fastethernet 4/0/0.1`	Specifies the subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.
Step 4	**encapsulation dot1q** *vlan-id* **second dot1q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1q 100 second dot1q 200`	Defines the matching criteria to map QinQ ingress frames on an interface to the appropriate service instance.
Step 5	**no ip address** **Example:**	Disables IP processing.

	Command or Action	Purpose
	`Device(config-subif)# no ip address`	
Step 6	**no shutdown** **Example:** `Device(config-subif)# no shutdown`	Restarts the Fast Ethernet subinterface.
Step 7	**exit** **Example:** `Device(config-subif)# exit`	Exits subinterface configuration mode and returns to global configuration mode.
Step 8	**template type pseudowire** *name* **Example:** `Device(config)# template type pseudowire temp4`	Creates a template pseudowire with a name that you specify and enters template configuration mode.
Step 9	**encapsulation mpls** **Example:** `Device(config-template)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 10	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to global configuration mode.
Step 11	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 109`	Establishes an interface pseudowire with a value that you specify and enters interface configuration mode.
Step 12	**source template type pseudowire** *name* **Example:** `Device(config-if)# source template type pseudowire temp4`	Configures the source template of type pseudowire named temp4.
Step 13	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 14	**neighbor** *peer-address vc id-value* **Example:** `Device(config-if)# neighbor 10.0.0.15 109`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 15	**signaling protocol ldp** **Example:** `Device(config-if)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.

	Command or Action	Purpose
Step 16	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts the interface pseudowire.
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 18	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con2`	Creates an L2VPN cross-connect context and enters xconnect configuration mode.
Step 19	**interworking ethernet** **Example:** `Device(config-xconnect)# interworking ethernet`	Specifies Ethernet as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 20	**member** *interface-type-number* **Example:** `Device(config-xconnect)# member fastethernet 4/0/0.1`	Specifies the location of the member interface.
Step 21	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 109`	Specifies a member pseudowire to form an L2VPN cross connect.
Step 22	**no shutdown** **Example:** `Device(config-xconnect)# no shutdown`	Restarts the member interface.
Step 23	**end** **Example:** `Device(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Routed Interworking on a HDLC PE Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation mpls**
5. **interworking ip**
6. **interface** *type slot/subslot /port* [**.** *subinterface*]
7. **no ip address** [*ip-address mask*] [**secondary**]

8. **xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class pw-iw-ip`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**interworking ip** **Example:** `Device(config-pw-class)# interworking ip`	Specifies IP as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 6	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config-pw-class)# interface serial 3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 7	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 8	**xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*] **Example:** `Device(config-if)# xconnect 198.51.100.2 123 pseudowire-class pw-iw-ip`	Creates the virtual circuit (VC) to transport the Layer 2 packets.
Step 9	**end** **Example:**	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

HDLC-to-Ethernet Routed Interworking on a HDLC PE Device Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template type pseudowire** *name*
4. **encapsulation mpls**
5. **exit**
6. **interface pseudowire** *number*
7. **source template type pseudowire** *name*
8. **encapsulation mpls**
9. **neighbor** *peer-address vc id-value*
10. **signaling protocol ldp**
11. **no shutdown**
12. **exit**
13. **l2vpn xconnect context** *context-name*
14. **interworking ip**
15. **member** *interface-type-number*
16. **member pseudowire** *interface-number*
17. **no shutdown**
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** *name* **Example:** `Device# template type pseudowire temp5`	Creates a template pseudowire with a name that you specify and enters template configuration mode.
Step 4	**encapsulation mpls** **Example:**	Specifies the tunneling encapsulation as MPLS.

	Command or Action	Purpose
	`Device(config-template)# encapsulation mpls`	
Step 5	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to global configuration mode.
Step 6	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 107`	Establishes an interface pseudowire with a value that you specify and enters interface configuration mode.
Step 7	**source template type pseudowire** *name* **Example:** `Device(config-if)# source template type pseudowire temp5`	Configures the source template of type pseudowire named temp5.
Step 8	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 9	**neighbor** *peer-address vc id-value* **Example:** `Device(config-if)# neighbor 10.0.0.11 107`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 10	**signaling protocol ldp** **Example:** `Device(config-if)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.
Step 11	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts the interface pseudowire.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 13	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates an L2VPN cross-connect context and enters xconnect configuration mode.
Step 14	**interworking ip** **Example:** `Device(config-xconnect)# interworking ip`	Specifies IP as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.

	Command or Action	Purpose
Step 15	**member** *interface-type-number* **Example:** `Device(config-xconnect)# member serial 0/1/0:0`	Specifies the location of the member interface.
Step 16	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 107`	Specifies a member pseudowire to form an L2VPN cross connect.
Step 17	**no shutdown** **Example:** `Device(config-xconnect)# no shutdown`	Restarts the member interface.
Step 18	**end** **Example:** `Device(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Routed Interworking (Port Mode) on an Ethernet PE Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation mpls**
5. **interworking ip**
6. **interface** *type slot/subslot /port* [**.** *subinterface*]
7. **encapsulation mpls**
8. **xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** [*pw-class-name*] **Example:**	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.

	Command or Action	Purpose
	`Device(config)# pseudowire-class pw-iw-ip`	
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**interworking ip** **Example:** `Device(config-pw-class)# interworking ip`	Specifies IP as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 6	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config-pw-class)# interface gigabitethernet 4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.
Step 7	**encapsulation mpls** **Example:** `Device(config-subif)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 8	**xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*] **Example:** `Device(config-subif)# xconnect 198.51.100.2 123 pseudowire-class pw-iw-ip`	Creates the virtual circuit (VC) to transport the Layer 2 packets.
Step 9	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Routed Interworking (Port Mode) on an Ethernet PE Device Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot*/*subslot* /*port* [**.** *subinterface*]
4. **encapsulation mpls**
5. **no ip address**
6. **no shutdown**

7. **exit**
8. **template type pseudowire** *name*
9. **encapsulation mpls**
10. **exit**
11. **interface pseudowire** *number*
12. **source template type pseudowire** *name*
13. **encapsulation mpls**
14. **neighbor** *peer-address vc id-value*
15. **signaling protocol ldp**
16. **no shutdown**
17. **exit**
18. **l2vpn xconnect context** *context-name*
19. **interworking ip**
20. **member** *interface-type-number*
21. **member pseudowire** *interface-number*
22. **no shutdown**
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config)# interface fastethernet 4/0/0.1`	Specifies the Fast Ethernet subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.
Step 4	**encapsulation mpls** **Example:** `Device(config-subif)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**no ip address** **Example:** `Device(config-subif)# no ip address`	Disables IP processing.

	Command or Action	Purpose
Step 6	**no shutdown** **Example:** `Device(config-subif)# no shutdown`	Restarts the Fast Ethernet subinterface.
Step 7	**exit** **Example:** `Device(config-subif)# exit`	Exits subinterface configuration mode and returns to global configuration mode.
Step 8	**template type pseudowire** *name* **Example:** `Device(config)# template type pseudowire temp4`	Creates a template pseudowire with a name that you specify and enters template configuration mode.
Step 9	**encapsulation mpls** **Example:** `Device(config-template)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 10	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to global configuration mode.
Step 11	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 109`	Establishes an interface pseudowire with a value that you specify and enters interface configuration mode.
Step 12	**source template type pseudowire** *name* **Example:** `Device(config-if)# source template type pseudowire temp4`	Configures the source template of type pseudowire named temp4.
Step 13	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 14	**neighbor** *peer-address vc id-value* **Example:** `Device(config-if)# neighbor 10.0.0.15 109`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 15	**signaling protocol ldp** **Example:** `Device(config-if)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.
Step 16	**no shutdown** **Example:**	Restarts the interface pseudowire.

	Command or Action	Purpose
	Device(config-if)# no shutdown	
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 18	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con2`	Creates an L2VPN cross-connect context and enters xconnect configuration mode.
Step 19	**interworking ip** **Example:** `Device(config-xconnect)# interworking ip`	Specifies IP as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 20	**member** *interface-type-number* **Example:** `Device(config-xconnect)# member fastethernet 4/0/0.1`	Specifies the location of the member interface.
Step 21	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 109`	Specifies a member pseudowire to form an L2VPN cross connect.
Step 22	**no shutdown** **Example:** `Device(config-xconnect)# no shutdown`	Restarts the member interface.
Step 23	**end** **Example:** `Device(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Routed Interworking (dot1q and QinQ Modes) on an Ethernet PE Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** [*pw-class-name*]
4. **encapsulation mpls**
5. **interworking ip**
6. **interface** *type slot/subslot /port* [**.** *subinterface*]
7. **encapsulation dot1q** *vlan-id* **second dot1q** *vlan-id*
8. **xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class pw-iw-ip`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 5	**interworking ip** **Example:** `Device(config-pw-class)# interworking ip`	Specifies IP as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 6	**interface** *type slot*/*subslot* /*port* [**.** *subinterface*] **Example:** `Device(config-pw-class)# interface gigabitethernet 4/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.
Step 7	**encapsulation dot1q** *vlan-id* **second dot1q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1q 100 second dot1q 200`	Defines the matching criteria to map QinQ ingress frames on an interface to the appropriate service instance.
Step 8	**xconnect** *peer-router-id vc id* **pseudowire-class** [*pw-class-name*] **Example:** `Device(config-subif)# xconnect 198.51.100.2 123 pseudowire-class pw-iw-ip`	Creates the virtual circuit (VC) to transport the Layer 2 packets.
Step 9	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

HDLC-to-Ethernet Routed Interworking (dot1q and QinQ Modes) on an Ethernet PE Device Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot/subslot /port* [**.** *subinterface*]
4. **encapsulation dot1q** *vlan-id* **second dot1q** *vlan-id*
5. **no ip address**
6. **no shutdown**
7. **exit**
8. **template type pseudowire** *name*
9. **encapsulation mpls**
10. **exit**
11. **interface pseudowire** *number*
12. **source template type pseudowire** *name*
13. **encapsulation mpls**
14. **neighbor** *peer-address vc id-value*
15. **signaling protocol ldp**
16. **no shutdown**
17. **exit**
18. **l2vpn xconnect context** *context-name*
19. **interworking ip**
20. **member** *interface-type-number*
21. **member pseudowire** *interface-number*
22. **no shutdown**
23. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot/subslot /port* [**.** *subinterface*] **Example:** `Device(config)# interface fastethernet 4/0/0.1`	Specifies the subinterface and enters subinterface configuration mode. • Ensure that the subinterface on the adjoining Ethernet CE device is on the same VLAN as this Ethernet PE device.

	Command or Action	Purpose
Step 4	**encapsulation dot1q** *vlan-id* **second dot1q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1q 100 second dot1q 200`	Defines the matching criteria to map QinQ ingress frames on an interface to the appropriate service instance.
Step 5	**no ip address** **Example:** `Device(config-subif)# no ip address`	Disables IP processing.
Step 6	**no shutdown** **Example:** `Device(config-subif)# no shutdown`	Restarts the Fast Ethernet subinterface.
Step 7	**exit** **Example:** `Device(config-subif)# exit`	Exits subinterface configuration mode and returns to global configuration mode.
Step 8	**template type pseudowire** *name* **Example:** `Device(config)# template type pseudowire temp4`	Creates a template pseudowire with a name that you specify and enters template configuration mode.
Step 9	**encapsulation mpls** **Example:** `Device(config-template)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 10	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to global configuration mode.
Step 11	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 109`	Establishes an interface pseudowire with a value that you specify and enters interface configuration mode.
Step 12	**source template type pseudowire** *name* **Example:** `Device(config-if)# source template type pseudowire temp4`	Configures the source template of type pseudowire named temp4.
Step 13	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.

	Command or Action	Purpose
Step 14	**neighbor** *peer-address vc id-value* **Example:** `Device(config-if)# neighbor 10.0.0.15 109`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 15	**signaling protocol ldp** **Example:** `Device(config-if)# signaling protocol ldp`	Specifies that the Label Distribution Protocol (LDP) is configured for the pseudowire class.
Step 16	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts the interface pseudowire.
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 18	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con2`	Creates an L2VPN cross-connect context and enters xconnect configuration mode.
Step 19	**interworking ip** **Example:** `Device(config-xconnect)# interworking ip`	Specifies IP as the type of pseudowire as well as the type of traffic that can flow across the pseudowire.
Step 20	**member** *interface-type-number* **Example:** `Device(config-xconnect)# member fastethernet 4/0/0.1`	Specifies the location of the member interface.
Step 21	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 109`	Specifies a member pseudowire to form an L2VPN cross connect.
Step 22	**no shutdown** **Example:** `Device(config-xconnect)# no shutdown`	Restarts the member interface.
Step 23	**end** **Example:** `Device(config-xconnect)# end`	Exits xconnect configuration mode and returns to privileged EXEC mode.

Verifying HDLC-to-Ethernet Interworking (Port Mode) Configuration on a HDLC PE Device

You can use **show** commands to view information about a HDLC-to-Ethernet interworking (port mode) configuration on a HDLC provider edge (PE) device.

SUMMARY STEPS

1. **show mpls l2transport vc**
2. **show mpls l2transport vc detail**
3. **show l2vpn atom vc**
4. **show l2vpn atom vc detail**

DETAILED STEPS

Step 1 **show mpls l2transport vc**

The following is sample output from the **show mpls l2transport vc** command which displays basic information about HDLC-to-Ethernet interworking (port mode) configuration on a HDLC PE device:

Example:

```
Device# show mpls l2transport vc

Local intf  Local circuit  Dest address    VC ID      Status
----------- -------------- --------------- ---------- ----------
Se0/1/0:0   HDLC           10.0.0.1        101        UP
```

Step 2 **show mpls l2transport vc detail**

The following is sample output from the **show mpls l2transport vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (port mode) configuration on a HDLC PE device:

Example:

```
Device# show mpls l2transport vc detail

Local interface: Se0/1/0:0 up, line protocol up, HDLC up
Interworking type is Ethernet
Destination address: 10.0.0.1, VC ID: 101, VC status: up
Output interface: Fa0/0/1, imposed label stack {20 22}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.10
Create time: 00:00:19, last status change time: 00:00:15
Last label FSM state change time: 00:00:15
Signaling protocol: LDP, peer 10.0.0.1:0 up
Targeted Hello: 203.0.113.1(LDP Id) -> 10.0.0.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Last local dataplane status rcvd: No fault
Last BFD dataplane status rcvd: Not sent
Last BFD peer monitor status rcvd: No fault
Last local AC circuit status rcvd: No fault
Last local AC circuit status sent: No fault
Last local PW i/f circ status rcvd: No fault
Last local LDP TLV status sent: No fault
Last remote LDP TLV status rcvd: No fault
```

```
Last remote LDP ADJ status rcvd: No fault
MPLS VC labels: local 33, remote 22
Group ID: local 0, remote 0
MTU: local 1500, remote 1500
Remote interface description: Connect to CE2
Sequencing: receive disabled, send disabled
Control Word: On
SSO Descriptor: 10.0.0.1/101, local label: 33
Dataplane:
SSM segment/switch IDs: 4274/4273 (used), PWID: 26
VC statistics:
transit packet totals: receive 3, send 6
transit byte totals: receive 162, send 366
transit packet drops: receive 0, seq error 0, send 0
```

Step 3 **show l2vpn atom vc**

The following is sample output from the **show l2vpn atom vc** command which displays basic information about HDLC-to-Ethernet interworking (port mode) configuration on a HDLC PE device:

Example:

```
Device# show l2vpn atom vc

                            Service
Interface Peer ID    VC ID  Type   Name  Status
--------- ---------- ------ ------ ----- ----------
pw101     10.0.0.1   101    p2p    101   UP
```

Step 4 **show l2vpn atom vc detail**

The following is sample output from the **show l2vpn atom vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (port mode) configuration on a HDLC PE device:

Example:

```
Device# show l2vpn atom vc detail

pseudowire101 is up, VC status is up PW type: Ethernet
Create time: 00:00:18, last status change time: 00:00:14
Last label FSM state change time: 00:00:14
Destination address: 10.0.0.1 VC ID: 101
Output interface: Fa0/0/1, imposed label stack {16 17}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.10
Member of xconnect service hdlc101
Associated member Se0/1/0:0 is up, status is up
Interworking type is Ethernet
Service id: 0xde000002
Signaling protocol: LDP, peer 10.0.0.1:0 up
Targeted Hello: 203.0.113.1(LDP Id) -> 10.0.0.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
PWid FEC (128), VC ID: 101
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Local dataplane status received : No fault
BFD dataplane status received : Not sent
BFD peer monitor status received : No fault
Status received from access circuit : No fault
Status sent to access circuit : No fault
Status received from pseudowire i/f : No fault
Status sent to network peer : No fault
```

```
Status received from network peer : No fault
Adjacency status of remote peer : No fault
Sequencing: receive disabled, send disabled
Bindings
Parameter    Local                          Remote
------------ ------------------------------ ------------------------------
Label        18                             17
Group ID     0                              0
Interface    Connect to CE1                 Connect to CE2
MTU          1500                           1500
Control word on (configured: autosense)     on
PW type      Ethernet                       Ethernet
VCCV CV type 0x02                           0x02
             LSPV [2]                       LSPV [2]
VCCV CC type 0x07                           0x07
             CW [1], RA [2], TTL [3]        CW [1], RA [2], TTL [3]
Status TLV   enabled                        supported
SSO Descriptor: 10.0.0.1/101, local label: 18
Dataplane:
SSM segment/switch IDs: 4106/4105 (used), PWID: 2
Rx Counters
3 input transit packets, 162 bytes
0 drops, 0 seq err
Tx Counters
5 output transit packets, 305 bytes
0 drops
```

Verifying HDLC-to-Ethernet Interworking (Port Mode) Configuration on an Ethernet PE Device

You can use **show** commands to view information about a HDLC-to-Ethernet interworking (port mode) configuration on an Ethernet PE device.

SUMMARY STEPS

1. **show mpls l2transport vc**
2. **show l2vpn atom vc**
3. **show l2vpn atom vc detail**

DETAILED STEPS

Step 1 **show mpls l2transport vc**

The following is sample output from the **show mpls l2transport vc** command which displays basic information about HDLC-to-Ethernet interworking (port mode) configuration on an Ethernet PE device:

Example:

```
Device# show mpls l2transport vc

Local interface: Gi1/0/0 up, line protocol up, Ethernet up
Destination address: 203.0.113.1, VC ID: 101, VC status: up
Output interface: Fa0/0/1, imposed label stack {19 33}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.11
Create time: 00:00:22, last status change time: 00:00:19
Last label FSM state change time: 00:00:19
```

```
Signaling protocol: LDP, peer 203.0.113.1:0 up
Targeted Hello: 10.0.0.1(LDP Id) -> 203.0.113.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Last local dataplane status rcvd: No fault
Last BFD dataplane status rcvd: Not sent
Last BFD peer monitor status rcvd: No fault
Last local AC circuit status rcvd: No fault
Last local AC circuit status sent: No fault
Last local PW i/f circ status rcvd: No fault
Last local LDP TLV status sent: No fault
Last remote LDP TLV status rcvd: No fault
Last remote LDP ADJ status rcvd: No fault
MPLS VC labels: local 22, remote 33
Group ID: local 0, remote 0
MTU: local 1500, remote 1500
Remote interface description: Connect to CE1
Sequencing: receive disabled, send disabled
Control Word: On
SSO Descriptor: 203.0.113.1/101, local label: 22
Dataplane:
SSM segment/switch IDs: 4574/4573 (used), PWID: 80
VC statistics:
transit packet totals: receive 9, send 5
transit byte totals: receive 315, send 380
transit packet drops: receive 0, seq error 0, send 0
```

Step 2 **show l2vpn atom vc**

The following is sample output from the **show l2vpn atom vc** command which displays basic information about HDLC-to-Ethernet interworking (port mode) configuration on an Ethernet PE device:

Example:

```
Device# show l2vpn atom vc

                            Service
Interface Peer ID    VC ID  Type   Name  Status
--------- ---------- ------ ------ ----- ----------
pw101     10.0.0.1   101    p2p    101   UP
```

Step 3 **show l2vpn atom vc detail**

The following is sample output from the **show l2vpn atom vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (port mode) configuration on an Ethernet PE device:

Example:

```
Device# show l2vpn atom vc detail

pseudowire101 is up, VC status is up PW type: Ethernet
Create time: 00:00:18, last status change time: 00:00:14
Last label FSM state change time: 00:00:14
Destination address: 10.0.0.1 VC ID: 101
Output interface: Fa0/0/1, imposed label stack {16 17}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.10
Member of xconnect service eth101
Associated member Se0/1/0:0 is up, status is up
Interworking type is Ethernet
Service id: 0xde000002
```

```
Signaling protocol: LDP, peer 10.0.0.1:0 up
Targeted Hello: 203.0.113.1(LDP Id) -> 10.0.0.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
PWid FEC (128), VC ID: 101
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Local dataplane status received : No fault
BFD dataplane status received : Not sent
BFD peer monitor status received : No fault
Status received from access circuit : No fault
Status sent to access circuit : No fault
Status received from pseudowire i/f : No fault
Status sent to network peer : No fault
Status received from network peer : No fault
Adjacency status of remote peer : No fault
Sequencing: receive disabled, send disabled
Bindings
Parameter    Local                          Remote
------------ ------------------------------ ------------------------------
Label        18                             17
Group ID     0                              0
Interface    Connect to CE1                 Connect to CE2
MTU          1500                           1500
Control word on (configured: autosense)     on
PW type      Ethernet                       Ethernet
VCCV CV type 0x02                           0x02
             LSPV [2]                       LSPV [2]
VCCV CC type 0x07                           0x07
             CW [1], RA [2], TTL [3]        CW [1], RA [2], TTL [3]
Status TLV   enabled                        supported
SSO Descriptor: 10.0.0.1/101, local label: 18
Dataplane:
SSM segment/switch IDs: 4106/4105 (used), PWID: 2
Rx Counters
3 input transit packets, 162 bytes
0 drops, 0 seq err
Tx Counters
5 output transit packets, 305 bytes
0 drops
```

Verifying HDLC-to-Ethernet Interworking (dot1q Mode) Configuration on a HDLC PE Device

You can use **show** commands to view information about a HDLC-to-Ethernet interworking (dot1q mode) configuration on a HDLC PE device.

SUMMARY STEPS

1. **show mpls l2transport vc**
2. **show mpls l2transport vc detail**
3. **show l2vpn atom vc**
4. **show l2vpn atom vc detail**

DETAILED STEPS

Step 1 **show mpls l2transport vc**

The following is sample output from the **show mpls l2transport vc** command which displays basic information about HDLC-to-Ethernet interworking (dot1q mode) configuration on a HDLC PE device:

Example:

```
Device# show mpls l2transport vc

Local intf  Local circuit  Dest address    VC ID      Status
----------- -------------- --------------- ---------- ----------
Se0/1/0:0   HDLC           10.0.0.1       101        UP
```

Step 2 **show mpls l2transport vc detail**

The following is sample output from the **show mpls l2transport vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (dot1q mode) configuration on a HDLC PE device:

Example:

```
Device# show mpls l2transport vc detail

Local interface: Se0/1/0:0 up, line protocol up, HDLC up
Interworking type is Ethernet
Destination address: 10.0.0.1, VC ID: 101, VC status: up
Output interface: Fa0/0/1, imposed label stack {20 22}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.10
Create time: 00:00:19, last status change time: 00:00:15
Last label FSM state change time: 00:00:15
Signaling protocol: LDP, peer 10.0.0.1:0 up
Targeted Hello: 203.0.113.1(LDP Id) -> 10.0.0.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Last local dataplane status rcvd: No fault
Last BFD dataplane status rcvd: Not sent
Last BFD peer monitor status rcvd: No fault
Last local AC circuit status rcvd: No fault
Last local AC circuit status sent: No fault
Last local PW i/f circ status rcvd: No fault
Last local LDP TLV status sent: No fault
Last remote LDP TLV status rcvd: No fault
Last remote LDP ADJ status rcvd: No fault
MPLS VC labels: local 33, remote 22
Group ID: local 0, remote 0
MTU: local 1500, remote 1500
Remote interface description: Connect to CE2
Sequencing: receive disabled, send disabled
Control Word: On
SSO Descriptor: 10.0.0.1/101, local label: 33
Dataplane:
SSM segment/switch IDs: 4274/4273 (used), PWID: 26
VC statistics:
transit packet totals: receive 3, send 6
transit byte totals: receive 162, send 366
transit packet drops: receive 0, seq error 0, send 0
```

Step 3 **show l2vpn atom vc**

The following is sample output from the **show l2vpn atom vc** command which displays basic information about HDLC-to-Ethernet interworking (dot1q mode) configuration on a HDLC PE device:

Example:

```
Device# show l2vpn atom vc

                            Service
Interface Peer ID    VC ID  Type   Name  Status
--------- ---------- ------ ------ ----- ----------
pw101     10.0.0.1  101    p2p    101   UP
```

Step 4 **show l2vpn atom vc detail**

The following is sample output from the **show l2vpn atom vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (dot1q mode) configuration on a HDLC PE device:

Example:

```
Device# show l2vpn atom vc detail

pseudowire101 is up, VC status is up PW type: Ethernet
Create time: 00:00:18, last status change time: 00:00:14
Last label FSM state change time: 00:00:14
Destination address: 10.0.0.1 VC ID: 101
Output interface: Fa0/0/1, imposed label stack {16 17}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.10
Member of xconnect service hdlc101
Associated member Se0/1/0:0 is up, status is up
Interworking type is Ethernet
Service id: 0xde000002
Signaling protocol: LDP, peer 10.0.0.1:0 up
Targeted Hello: 203.0.113.1(LDP Id) -> 10.0.0.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
PWid FEC (128), VC ID: 101
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Local dataplane status received : No fault
BFD dataplane status received : Not sent
BFD peer monitor status received : No fault
Status received from access circuit : No fault
Status sent to access circuit : No fault
Status received from pseudowire i/f : No fault
Status sent to network peer : No fault
Status received from network peer : No fault
Adjacency status of remote peer : No fault
Sequencing: receive disabled, send disabled
Bindings
Parameter    Local                          Remote
------------ ------------------------------ ------------------------------
Label        18                             17
Group ID     0                              0
Interface    Connect to CE1                 Connect to CE2
MTU          1500                           1500
Control word on (configured: autosense)     on
PW type      Ethernet                       Ethernet
VCCV CV type 0x02                           0x02
             LSPV [2]                       LSPV [2]
VCCV CC type 0x07                           0x07
```

```
             CW [1], RA [2], TTL [3]        CW [1], RA [2], TTL [3]
Status TLV   enabled                        supported
SSO Descriptor: 10.0.0.1/101, local label: 18
Dataplane:
SSM segment/switch IDs: 4106/4105 (used), PWID: 2
Rx Counters
3 input transit packets, 162 bytes
0 drops, 0 seq err
Tx Counters
5 output transit packets, 305 bytes
0 drops
```

Verifying HDLC-to-Ethernet Interworking (dot1q Mode) Configuration on an Ethernet PE Device

You can use **show** commands to view information about a HDLC-to-Ethernet interworking (dot1q mode) configuration on an Ethernet PE device.

SUMMARY STEPS

1. **show mpls l2transport vc**
2. **show mpls l2transport vc detail**
3. **show l2vpn atom vc**
4. **show l2vpn atom vc detail**

DETAILED STEPS

Step 1 **show mpls l2transport vc**

The following is sample output from the **show mpls l2transport vc** command which displays basic information about HDLC-to-Ethernet interworking (dot1q mode) configuration on an Ethernet PE device:

Example:

```
Device# show mpls l2transport vc

Local intf  Local circuit  Dest address    VC ID      Status
----------- -------------- --------------- ---------- ----------
Gi1/0/0.10  Eth VLAN 10    203.0.113.1     138        UP
```

Step 2 **show mpls l2transport vc detail**

The following is sample output from the **show mpls l2transport vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (dot1q mode) configuration on an Ethernet PE device:

Example:

```
Device# show mpls l2transport vc detail

Local interface: Gi1/0/0.10 up, line protocol up, Eth VLAN 10 up
Interworking type is Ethernet
Destination address: 203.0.113.1, VC ID: 138, VC status: up
Output interface: Fa0/0/1, imposed label stack {19 35}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.11
Create time: 00:00:22, last status change time: 00:00:20
Last label FSM state change time: 00:00:20
```

```
Signaling protocol: LDP, peer 203.0.113.1:0 up
Targeted Hello: 10.0.0.1(LDP Id) -> 203.0.113.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Last local dataplane status rcvd: No fault
Last BFD dataplane status rcvd: Not sent
Last BFD peer monitor status rcvd: No fault
Last local AC circuit status rcvd: No fault
Last local AC circuit status sent: No fault
Last local PW i/f circ status rcvd: No fault
Last local LDP TLV status sent: No fault
Last remote LDP TLV status rcvd: No fault
Last remote LDP ADJ status rcvd: No fault
MPLS VC labels: local 53, remote 35
Group ID: local 0, remote 0
MTU: local 1500, remote 1500
Remote interface description: Connect to CE1
Sequencing: receive disabled, send disabled
Control Word: On
SSO Descriptor: 203.0.113.1/138, local label: 53
Dataplane:
SSM segment/switch IDs: 4784/4783 (used), PWID: 117
VC statistics:
transit packet totals: receive 6, send 6
transit byte totals: receive 234, send 1276
transit packet drops: receive 0, seq error 0, send 0
```

Step 3 **show l2vpn atom vc**

The following is sample output from the **show l2vpn atom vc** command which displays basic information about HDLC-to-Ethernet interworking (dot1q mode) configuration on an Ethernet PE device:

Example:

```
Device# show l2vpn atom vc

                            Service
Interface Peer ID    VC ID  Type   Name  Status
--------- ---------- ------ ------ ----- ----------
pw138     203.0.113.1 138    p2p    138   UP
```

Step 4 **show l2vpn atom vc detail**

The following is sample output from the **show l2vpn atom vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (dot1q mode) configuration on an Ethernet PE device:

Example:

```
Device# show l2vpn atom vc detail

pseudowire138 is up, VC status is up PW type: Ethernet
Create time: 00:00:23, last status change time: 00:00:20
Last label FSM state change time: 00:00:20
Destination address: 203.0.113.1 VC ID: 138
Output interface: Fa0/0/1, imposed label stack {18 20}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.11
Member of xconnect service eth138
Associated member Gi1/0/0.10 is up, status is up
Interworking type is Ethernet
Service id: 0x7b000029
```

```
Signaling protocol: LDP, peer 203.0.113.1:0 up
Targeted Hello: 10.0.0.1(LDP Id) -> 203.0.113.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
PWid FEC (128), VC ID: 138
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Local dataplane status received : No fault
BFD dataplane status received : Not sent
BFD peer monitor status received : No fault
Status received from access circuit : No fault
Status sent to access circuit : No fault
Status received from pseudowire i/f : No fault
Status sent to network peer : No fault
Status received from network peer : No fault
Adjacency status of remote peer : No fault
Sequencing: receive disabled, send disabled
Bindings
Parameter    Local                          Remote
------------ ------------------------------ ------------------------------
Label        30                             20
Group ID     0                              0
Interface    Connect to CE2                 Connect to CE1
MTU          1500                           1500
Control word on (configured: autosense)     on
PW type      Ethernet                       Ethernet
VCCV CV type 0x02                           0x02
             LSPV [2]                       LSPV [2]
VCCV CC type 0x07                           0x07
             CW [1], RA [2], TTL [3]        CW [1], RA [2], TTL [3]
Status TLV   enabled                        supported
SSO Descriptor: 203.0.113.1/138, local label: 30
Dataplane:
SSM segment/switch IDs: 4333/4332 (used), PWID: 41
Rx Counters
8 input transit packets, 312 bytes
0 drops, 0 seq err
Tx Counters
5 output transit packets, 380 bytes
0 drops
```

Verifying HDLC-to-Ethernet Interworking (QinQ Mode) Configuration on a HDLC PE Device

You can use **show** commands to view information about a HDLC-to-Ethernet interworking (QinQ mode) configuration on a HDLC PE device.

SUMMARY STEPS

1. **show mpls l2transport vc**
2. **show mpls l2transport vc detail**
3. **show l2vpn atom vc**
4. **show l2vpn atom vc detail**

DETAILED STEPS

Step 1 **show mpls l2transport vc**

The following is sample output from the **show mpls l2transport vc** command which displays basic information about HDLC-to-Ethernet interworking (QinQ mode) configuration on a HDLC PE device:

Example:

```
Device# show mpls l2transport vc

Local intf  Local circuit  Dest address    VC ID      Status
----------- -------------- --------------- ---------- ----------
Se0/1/0:0   HDLC           10.0.0.1        145        UP
```

Step 2 **show mpls l2transport vc detail**

The following is sample output from the **show mpls l2transport vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (QinQ mode) configuration on a HDLC PE device:

Example:

```
Device# show mpls l2transport vc detail

Local interface: Se0/1/0:0 up, line protocol up, HDLC up
Interworking type is Ethernet
Destination address: 10.0.0.1, VC ID: 101, VC status: up
Output interface: Fa0/0/1, imposed label stack {20 22}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.10
Create time: 00:00:19, last status change time: 00:00:15
Last label FSM state change time: 00:00:15
Signaling protocol: LDP, peer 10.0.0.1:0 up
Targeted Hello: 203.0.113.1(LDP Id) -> 10.0.0.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Last local dataplane status rcvd: No fault
Last BFD dataplane status rcvd: Not sent
Last BFD peer monitor status rcvd: No fault
Last local AC circuit status rcvd: No fault
Last local AC circuit status sent: No fault
Last local PW i/f circ status rcvd: No fault
Last local LDP TLV status sent: No fault
Last remote LDP TLV status rcvd: No fault
Last remote LDP ADJ status rcvd: No fault
MPLS VC labels: local 33, remote 22
Group ID: local 0, remote 0
MTU: local 1500, remote 1500
Remote interface description: Connect to CE2
Sequencing: receive disabled, send disabled
Control Word: On
SSO Descriptor: 10.0.0.1/101, local label: 33
Dataplane:
SSM segment/switch IDs: 4274/4273 (used), PWID: 26
VC statistics:
transit packet totals: receive 3, send 6
transit byte totals: receive 162, send 366
transit packet drops: receive 0, seq error 0, send 0
```

Step 3 **show l2vpn atom vc**

The following is sample output from the **show l2vpn atom vc** command which displays basic information about HDLC-to-Ethernet interworking (QinQ mode) configuration on a HDLC PE device:

Example:

```
Device# show l2vpn atom vc

                              Service
Interface Peer ID    VC ID  Type   Name  Status
--------- ---------- ------ ------ ----- ----------
pw145     10.0.0.1   145    p2p    145   UP
```

Step 4 **show l2vpn atom vc detail**

The following is sample output from the **show l2vpn atom vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (QinQ mode) configuration on a HDLC PE device:

Example:

```
Device# show l2vpn atom vc detail

pseudowire145 is up, VC status is up PW type: Ethernet
Create time: 00:00:18, last status change time: 00:00:13
Last label FSM state change time: 00:00:13
Destination address: 10.0.0.1 VC ID: 145
Output interface: Fa0/0/1, imposed label stack {16 33}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.10
Member of xconnect service hdlc145
Associated member Se0/1/0:0 is up, status is up
Interworking type is Ethernet
Service id: 0x2e
Signaling protocol: LDP, peer 10.0.0.1:0 up
Targeted Hello: 203.0.113.1(LDP Id) -> 10.0.0.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
PWid FEC (128), VC ID: 145
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Local dataplane status received : No fault
BFD dataplane status received : Not sent
BFD peer monitor status received : No fault
Status received from access circuit : No fault
Status sent to access circuit : No fault
Status received from pseudowire i/f : No fault
Status sent to network peer : No fault
Status received from network peer : No fault
Adjacency status of remote peer : No fault
Sequencing: receive disabled, send disabled
Bindings
Parameter    Local                          Remote
------------ ------------------------------ ------------------------------
Label        33                             33
Group ID     0                              0
Interface    Connect to CE1                 Connect to CE2
MTU          1500                           1500
Control word on (configured: autosense)     on
PW type      Ethernet                       Ethernet
VCCV CV type 0x02                           0x02
             LSPV [2]                       LSPV [2]
VCCV CC type 0x07                           0x07
```

```
             CW [1], RA [2], TTL [3]        CW [1], RA [2], TTL [3]
Status TLV   enabled                        supported
SSO Descriptor: 10.0.0.1/145, local label: 33
Dataplane:
SSM segment/switch IDs: 4345/4344 (used), PWID: 48
Rx Counters
2 input transit packets, 108 bytes
0 drops, 0 seq err
Tx Counters
3 output transit packets, 183 bytes
0 drops
```

Verifying HDLC-to-Ethernet Interworking (QinQ Mode) Configuration on an Ethernet PE Device

You can use **show** commands to view information about a HDLC-to-Ethernet interworking (QinQ mode) configuration on an Ethernet PE device.

SUMMARY STEPS

1. **show mpls l2transport vc**
2. **show mpls l2transport vc detail**
3. **show l2vpn atom vc**
4. **show l2vpn atom vc detail**

DETAILED STEPS

Step 1 **show mpls l2transport vc**

The following is sample output from the **show mpls l2transport vc** command which displays basic information about HDLC-to-Ethernet interworking (QinQ mode) configuration on an Ethernet PE device:

Example:

```
Device# show mpls l2transport vc

Local intf  Local circuit  Dest address    VC ID      Status
----------- -------------- --------------- ---------- ----------
Gi1/0/0.10  Eth VLAN 10/20  203.0.113.1    145        UP
```

Step 2 **show mpls l2transport vc detail**

The following is sample output from the **show mpls l2transport vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (QinQ mode) configuration on an Ethernet PE device:

Example:

```
Device# show mpls l2transport vc detail

Local interface: Gi1/0/0.10 up, line protocol up, Eth VLAN 10/20 up
Interworking type is Ethernet
Destination address: 203.0.113.1, VC ID: 145, VC status: up
Output interface: Fa0/0/1, imposed label stack {19 27}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.11
Create time: 00:00:23, last status change time: 00:00:21
Last label FSM state change time: 00:00:21
```

```
Signaling protocol: LDP, peer 203.0.113.1:0 up
Targeted Hello: 10.0.0.1(LDP Id) -> 203.0.113.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Last local dataplane status rcvd: No fault
Last BFD dataplane status rcvd: Not sent
Last BFD peer monitor status rcvd: No fault
Last local AC circuit status rcvd: No fault
Last local AC circuit status sent: No fault
Last local PW i/f circ status rcvd: No fault
Last local LDP TLV status sent: No fault
Last remote LDP TLV status rcvd: No fault
Last remote LDP ADJ status rcvd: No fault
MPLS VC labels: local 25, remote 27
Group ID: local 0, remote 0
MTU: local 1500, remote 1500
Remote interface description: Connect to CE1
Sequencing: receive disabled, send disabled
Control Word: On
SSO Descriptor: 203.0.113.1/145, local label: 25
Dataplane:
SSM segment/switch IDs: 4815/4814 (used), PWID: 124
VC statistics:
transit packet totals: receive 10, send 6
transit byte totals: receive 430, send 456
transit packet drops: receive 0, seq error 0, send 0
```

Step 3 **show l2vpn atom vc**

The following is sample output from the **show l2vpn atom vc** command which displays basic information about HDLC-to-Ethernet interworking (QinQ mode) configuration on an Ethernet PE device:

Example:

```
Device# show l2vpn atom vc

                            Service
Interface Peer ID    VC ID  Type   Name  Status
--------- ---------- ------ ------ ----- ----------
pw145     203.0.113.1  145    p2p    145   UP
```

Step 4 **show l2vpn atom vc detail**

The following is sample output from the **show l2vpn atom vc detail** command which displays detailed information about HDLC-to-Ethernet interworking (QinQ mode) configuration on an Ethernet PE device:

Example:

```
Device# show l2vpn atom vc detail

pseudowire145 is up, VC status is up PW type: Ethernet
Create time: 00:00:23, last status change time: 00:00:19
Last label FSM state change time: 00:00:19
Destination address: 203.0.113.1 VC ID: 145
Output interface: Fa0/0/1, imposed label stack {18 33}
Preferred path: not configured
Default path: active
Next hop: 10.0.0.11
Member of xconnect service eth145
Associated member Gi1/0/0.10 is up, status is up
Interworking type is Ethernet
Service id: 0xed000030
```

```
Signaling protocol: LDP, peer 203.0.113.1:0 up
Targeted Hello: 10.0.0.1(LDP Id) -> 203.0.113.1, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
PWid FEC (128), VC ID: 145
Status TLV support (local/remote) : enabled/supported
LDP route watch : enabled
Label/status state machine : established, LruRru
Local dataplane status received : No fault
BFD dataplane status received : Not sent
BFD peer monitor status received : No fault
Status received from access circuit : No fault
Status sent to access circuit : No fault
Status received from pseudowire i/f : No fault
Status sent to network peer : No fault
Status received from network peer : No fault
Adjacency status of remote peer : No fault
Sequencing: receive disabled, send disabled
Bindings
Parameter    Local                          Remote
------------ ------------------------------ ------------------------------
Label        33                             33
Group ID     0                              0
Interface    Connect to CE2                 Connect to CE1
MTU          1500                           1500
Control word on (configured: autosense)     on
PW type      Ethernet                       Ethernet
VCCV CV type 0x02                           0x02
             LSPV [2]                       LSPV [2]
VCCV CC type 0x07                           0x07
             CW [1], RA [2], TTL [3]        CW [1], RA [2], TTL [3]
Status TLV   enabled                        supported
SSO Descriptor: 203.0.113.1/145, local label: 33
Dataplane:
SSM segment/switch IDs: 4361/4360 (used), PWID: 48
Rx Counters
8 input transit packets, 344 bytes
0 drops, 0 seq err
Tx Counters
5 output transit packets, 380 bytes
0 drops
```

Verifying L2VPN Interworking

To verify the L2VPN status (in the AToM configuration), use the following commands:

- **show connection** [**all** | **name** | **id** | **elements** | **port**]
- **show xconnect** [**all** | **interface** | **peer**]
- **show mpls l2transport** [**binding** | **checkpoint** | **hw-capability** | **summary** | **vc**]
- **show mpls infrastructure lfd pseudowire vcid**

Verifying L2VPN Interworking using the commands associated with the L2VPN Protocol-Based CLIs feature

To verify the L2VPN status (in the AToM configuration), use the following commands:

- **show connection** [**all** | **name** | **id** | **elements** | **port**]
- **show l2vpn service**[**all** | **interface** | **peer**]
- **show l2vpn atom** [**binding** | **checkpoint** | **hw-capability** | **summary** | **vc**]
- **show mpls infrastructure lfd pseudowire vcid**

Configuration Examples for L2VPN Interworking

Frame Relay DLCI-to-Ethernet VLAN 802.1Q Using Bridged Internetworking Example

The following example shows how to configure the Frame Relay DLCI-to-Ethernet VLAN 802.1Q feature using bridged interworking:

PE1 router	PE2 router

`config t`
`mpls label protocol ldp`
`interface Loopback100`
` ip address 10.0.0.100 255.255.255.255`
`pseudowire-class fr-vlan`
` encapsulation mpls`
` interworking ethernet`
`frame-relay switching`
`interface serial 2/0/0:1`
` encapsulation frame-relay`
` frame-relay intf-type dce`
`connect mpls serial 2/0/0:1 567 l2transport`
 	
` xconnect 10.0.0.200 150 pw-class fr-vlan`
	`config t`
	`mpls label protocol ldp`
	`interface Loopback200`
	` ip address 10.0.0.200 255.255.255.255`
	`pseudowire-class fr-vlan`
	` encapsulation mpls`
	` interworking ethernet`
	`interface gigabitethernet 5/1/0.3`
	` encapsulation dot1q 1525`
	
	` xconnect 10.0.0.100 150 pw-class fr-vlan`

Frame Relay DLCI-to-Ethernet VLAN 802.1Q Using Bridged Internetworking Example using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows how to configure the Frame Relay DLCI-to-Ethernet VLAN 802.1Q feature using bridged interworking:

<table>
<tr><th>PE1 router</th><th>PE2 router</th></tr>
<tr><td><pre>config t
mpls label protocol ldp
interface Loopback100
 ip address 10.0.0.100 255.255.255.255
template type pseudowire fr-vlan
 encapsulation mpls
 interworking ethernet
frame-relay switching
interface serial 2/0/0:1
 encapsulation frame-relay
 frame-relay intf-type dce
connect mpls serial 2/0/0:1 567 l2transport

 interface pseudowire 100
 source template type pseudowire fr-vlan
 neighbor 10.0.0.200 150
!
l2vpn xconnect context con1
 member pseudowire 100
 member 10.0.0.200 150 encapsulation mpls</pre></td><td><pre>config t
mpls label protocol ldp
interface Loopback200
 ip address 10.0.0.200 255.255.255.255
template type pseudowire fr-vlan
 encapsulation mpls
 interworking ethernet
interface gigabitethernet 5/1/0.3
 encapsulation dot1q 1525

 interface pseudowire 100
 source template type pseudowire fr-vlan
 neighbor 10.0.0.100 150
!
l2vpn xconnect context con1
 member pseudowire 100
 member 10.0.0.100 150 encapsulation mpls</pre></td></tr>
</table>

ATM AAL5-to-Ethernet VLAN 802.1Q Using Bridged Internetworking Example

The following example shows how to configure the ATM AAL5-to-Ethernet VLAN 802.1Q feature using bridged interworking:

<table>
<tr><th>PE1 router</th><th>PE2 router</th></tr>
<tr><td><pre>config t
mpls label protocol ldp
interface Loopback100
 ip address 10.0.0.100 255.255.255.255
pseudowire-class atm-vlan
 encapsulation mpls
 interworking ethernet
interface atm 2/0/0
 pvc 0/200 l2transport
 encapsulation aal5snap

 xconnect 10.0.0.200 140 pw-class atm-vlan</pre></td><td><pre>config t
mpls label protocol ldp
interface Loopback200
 ip address 10.0.0.200 255.255.255.255
pseudowire-class atm-vlan
 encapsulation mpls
 interworking ethernet
interface gigabitethernet 5/1/0.3
 encapsulation dot1q 1525

 xconnect 10.0.0.100 140 pw-class atm-vlan</pre></td></tr>
</table>

ATM AAL5-to-Ethernet VLAN 802.1Q Using Bridged Internetworking Example using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows how to configure the ATM AAL5-to-Ethernet VLAN 802.1Q feature using bridged interworking:

PE1 router	PE2 router
config t mpls label protocol ldp interface Loopback100 ip address 10.0.0.100 255.255.255.255 template type pseudowire atm-vlan encapsulation mpls interworking ethernet interface atm 2/0/0 pvc 0/200 l2transport encapsulation aal5snap interface pseudowire 100 source template type pseudowire atm-vlan neighbor 10.0.0.200 140 ! l2vpn xconnect context con1 member pseudowire 100 member 10.0.0.200 140 encapsulation mpls	config t mpls label protocol ldp interface Loopback200 ip address 10.0.0.200 255.255.255.255 template type pseudowire atm-vlan encapsulation mpls interworking ethernet interface gigabitethernet 5/1/0.3 encapsulation dot1q 1525 interface pseudowire 100 source template type pseudowire atm-vlan neighbor 10.0.0.100 140 ! l2vpn xconnect context con1 member pseudowire 100 member 10.0.0.200 140 encapsulation mpls

ATM AAL5-to-Ethernet Port Using Routed Interworking Example

The following example shows how to configure the ATM AAL5-to-Ethernet Port feature using routed interworking:

PE1 router	PE2 router
config t mpls label protocol ldp interface Loopback100 ip address 10.0.0.100 255.255.255.255 pseudowire-class atm-eth encapsulation mpls interworking ip interface atm 2/0.1 pvc 0/200 l2transport encapsulation aal5 xconnect 10.0.0.200 140 pw-class atm-eth	config t mpls label protocol ldp interface Loopback200 ip address 10.0.0.200 255.255.255.255 pseudowire-class atm-eth encapsulation mpls interworking ip interface gigabitethernet 5/1/0 xconnect 10.0.0.100 140 pw-class atm-eth

Frame Relay DLCI-to-Ethernet Port Using Routed Interworking Example

The following example shows how to configure the Frame Relay DLCI-to-Ethernet Port feature using routed interworking:

PE1 router	PE2 router

`config t`
`mpls label protocol ldp`
`interface Loopback100`
` ip address 10.0.0.100 255.255.255.255`
`pseudowire-class fr-eth`
` encapsulation mpls`
` interworking ip`
`frame-relay switching`
`interface serial 2/0/0:1`
` encapsulation frame-relay`
` frame-relay intf-type dce`
` frame-relay interface-dlci 567 switched`
`connect fr-vlan-1 POS2/3/1 151 l2transport`
 	
` xconnect 10.0.0.200 151 pw-class pw-class-bridge`
	`config t`
	`mpls label protocol ldp`
	`interface Loopback200`
	` ip address 10.0.0.200 255.255.255.255`
	`pseudowire-class fr-eth`
	` encapsulation mpls`
	` interworking ip`
	`interface gigabitethernet 5/1/0`
	
	` xconnect 10.0.0.100 150 pw-class fr-eth`

Frame Relay DLCI-to-Ethernet Port Using Routed Interworking Example using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows how to configure the Frame Relay DLCI-to-Ethernet Port feature using routed interworking:

PE1 router	PE2 router

`config t`
`mpls label protocol ldp`
`interface Loopback100`
` ip address 10.0.0.100 255.255.255.255`
`template type pseudowire fr-eth`
` encapsulation mpls`
` interworking ip`
`frame-relay switching`
`interface serial 2/0/0:1`
` encapsulation frame-relay`
` frame-relay intf-type dce`
` frame-relay interface-dlci 567 switched`
`connect fr-vlan-1 POS2/3/1 151 l2transport`
 	
` interface pseudowire 100`
` source template type pseudowire fr-eth`
` neighbor 10.0.0.200 140`
`!`
`l2vpn xconnect context con1`
` member pseudowire 100`
` member 10.0.0.200 140 encapsulation mpls`
	`config t`
	`mpls label protocol ldp`
	`interface Loopback200`
	` ip address 10.0.0.200 255.255.255.255`
	`template type pseudowire fr-eth`
	` encapsulation mpls`
	` interworking ip`
	`interface gigabitethernet 5/1/0`
	
	` interface pseudowire 100`
	` source template type pseudowire fr-eth`
	` neighbor 10.0.0.200 140`
	`!`
	`l2vpn xconnect context con1`
	` member pseudowire 100`
	` member 10.0.0.200 140 encapsulation mpls`

Ethernet-to-VLAN over AToM--Bridged Example

The following example shows how to configure Ethernet-to-VLAN over AToM in a PE router:

PE1 router	PE2 router
ip cef ! mpls label protocol ldp mpls ldp router-id Loopback0 force ! pseudowire-class atom encapsulation mpls ! interface Loopback0 ip address 10.9.9.9 255.255.255.255 ! interface FastEthernet0/0 no ip address ! interface FastEthernet1/0 xconnect 10.8.8.8 123 pw-class atom	ip cef ! mpls label protocol ldp mpls ldp router-id Loopback0 force ! pseudowire-class atom-eth-iw encapsulation mpls interworking ethernet ! interface Loopback0 ip address 10.8.8.8 255.255.255.255 ! interface FastEthernet1/0.1 encapsulation dot1q 100 xconnect 10.9.9.9 123 pw-class atom-eth-iw

Ethernet to VLAN over AToM (Bridged) Example using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows the configuration of Ethernet to VLAN over AToM:

PE1	PE2
ip cef	ip cef
!	!
mpls label protocol ldp	mpls label protocol ldp
mpls ldp router-id Loopback0 force	mpls ldp router-id Loopback0 force
!	!
template type pseudowire atom-eth-iw	template type pseudowire atom
encapsulation mpls	encapsulation mpls
interworking ethernet	!
!	interface Loopback0
interface Loopback0	ip address 10.9.9.9 255.255.255.255
ip address 10.8.8.8 255.255.255.255	!
!	interface FastEthernet0/0
interface FastEthernet1/0.1	no ip address
encapsulation dot1q 100	!
interface pseudowire 100 source template type pseudowire atom-eth-iw	interface FastEthernet1/0
neighbor 10.8.8.8 123 ! l2vpn xconnect context con1 member pseudowire 100 member 10.8.8.8 123 encapsulation mpls	interface pseudowire 100 source template type pseudowire ether-pw neighbor 10.9.9.9 123 ! l2vpn xconnect context con1 member pseudowire 100 member 10.9.9.9 123 encapsulation mpls

VLAN-to-ATM AAL5 over AToM (Bridged) Example

The following example shows the configuration of VLAN-to-ATM AAL5 over AToM:

PE1 router	PE2 router

PE1 router	PE2 router
	ip cef ! mpls ip mpls label protocol ldp mpls ldp router-id Loopback0 ! pseudowire-class inter-ether encapsulation mpls interworking ethernet ! interface Loopback0 ip address 10.9.9.9 255.255.255.255 ! interface FastEthernet0/0 no ip address ! interface FastEthernet0/0.1 encapsulation dot1Q 10 xconnect 10.8.8.8 123 pw-class inter-ether ! router ospf 10 log-adjacency-changes network 10.9.9.9 0.0.0.0 area 0 network 10.1.1.2 0.0.0.0 area 0

PE1 router	PE2 router
ip cef ! mpls ip mpls label protocol ldp mpls ldp router-id Loopback0 ! pseudowire-class inter-ether encapsulation mpls interworking ethernet ! interface Loopback0 ip address 10.8.8.8 255.255.255.255 ! interface ATM1/0.1 point-to-point pvc 0/100 l2transport encapsulation aal5snap xconnect 10.9.9.9 123 pw-class inter-ether ! interface FastEthernet1/0 xconnect 10.9.9.9 1 pw-class inter-ether ! router ospf 10 log-adjacency-changes network 10.8.8.8 0.0.0.0 area 0	

PE1 router	PE2 router
`network 10.1.1.1 0.0.0.0 area 0`	

VLAN-to-ATM AAL5 over AToM (Bridged) Example using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows the configuration of VLAN-to-ATM AAL5 over AToM:

PE1 router	PE2 router

PE1 router	PE2 router
ip cef ! mpls ip mpls label protocol ldp mpls ldp router-id Loopback0 ! template type pseudowire inter-ether encapsulation mpls interworking ethernet ! interface Loopback0 ip address 10.8.8.8 255.255.255.255 ! interface ATM1/0.1 point-to-point pvc 0/100 l2transport encapsulation aal5snap interface pseudowire 100 source template type pseudowire inter-ether neighbor 10.9.9.9 123 ! l2vpn xconnect context con1 ! interface FastEthernet1/0	ip cef ! mpls ip mpls label protocol ldp mpls ldp router-id Loopback0 ! template type pseudowire inter-ether encapsulation mpls interworking ethernet ! interface Loopback0 ip address 10.9.9.9 255.255.255.255 ! interface FastEthernet0/0 no ip address ! interface FastEthernet0/0.1 encapsulation dot1Q 10 interface pseudowire 100 source template type pseudowire inter-ether neighbor 10.8.8.8 123 ! l2vpn xconnect context con1 member pseudowire 100

PE1 router	PE2 router
interface pseudowire 100 source template type pseudowire inter-ether neighbor 10.9.9.9 1 ! l2vpn xconnect context con1 member pseudowire 100 member 10.9.9.9.9 1 encapsulation mpls ! router ospf 10 log-adjacency-changes network 10.8.8.8 0.0.0.0 area 0 network 10.1.1.1 0.0.0.0 area 0	member 10.8.8.8 123 encapsulation mpls ! router ospf 10 log-adjacency-changes network 10.9.9.9 0.0.0.0 area 0 network 10.1.1.2 0.0.0.0 area 0

Ethernet VLAN-to-PPP over AToM (Routed) Example

The following example shows the configuration of Ethernet VLAN-to-PPP over AToM

PE1 router	PE2 router
configure terminal mpls label protocol ldp mpls ldp router-id Loopback0 mpls ip ! pseudowire-class ppp-ether encapsulation mpls interworking ip ! interface Loopback0 ip address 10.8.8.8 255.255.255.255 no shutdown ! interface POS2/0/1 no ip address encapsulation ppp no peer default ip address ppp ipcp address proxy 10.10.10.1 xconnect 10.9.9.9 300 pw-class ppp-ether no shutdown	configure terminal mpls label protocol ldp mpls ldp router-id Loopback0 mpls ip ! pseudowire-class ppp-ether encapsulation mpls interworking ip ! interface Loopback0 ip address 10.9.9.9 255.255.255.255 no shutdown ! interface GigabitEthernet6/2 xconnect 10.8.8.8 300 pw-class ppp-ether no shutdown

Ethernet VLAN to PPP over AToM (Routed) Example using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows the configuration of Ethernet VLAN to PPP over AToM:

PE1	PE2

PE1	PE2
configure terminal	configure terminal
mpls label protocol ldp	mpls label protocol ldp
mpls ldp router-id Loopback0	mpls ldp router-id Loopback0
mpls ip	mpls ip
!	!
template type pseudowire ppp-ether	template type pseudowire ppp-ether
encapsulation mpls	encapsulation mpls
interworking ip	interworking ip
!	!
interface Loopback0	interface Loopback0
ip address 10.8.8.8 255.255.255.255	ip address 10.9.9.9 255.255.255.255
no shutdown	no shutdown
!	!
interface POS2/0/1	interface vlan300
no ip address	mtu 4470
encapsulation ppp	no ip address
no peer default ip address	interface pseudowire 100
ppp ipcp address proxy 10.10.10.1	source template type pseudowire ppp-ether
interface pseudowire 100	neighbor 10.8.8.8 300
source template type pseudowire ppp-ether	!
neighbor 10.9.9.9 300	l2vpn xconnect context con1 member pseudowire 100 member 10.8.8.8 300 encapsulation mpls
!	no shutdown
l2vpn xconnect context con1 member pseudowire 100	

PE1	PE2
member 10.9.9.9 300 encapsulation mpls no shutdown	! interface GigabitEthernet6/2 switchport switchport trunk encapsulation dot1q switchport trunk allowed vlan 300 switchport mode trunk no shutdown

ATM VC-to-VC Local Switching (Different Port) Example

The following example shows the configuration of ATM VC-to-VC local switching:

CE1 router	CE2 router	PE router

interface ATM1/0
 no ip address
 atm clock INTERNAL
 no atm ilmi-keepalive
 no atm enable-ilmi-trap
 	
interface ATM1/0
 ip address 10.1.1.1 255.255.255.0
 no atm enable-ilmi-trap
 pvc 0/100
  encapsulation aal5snap
	interface ATM3/0
	 no ip address
	 atm clock INTERNAL
	 no atm ilmi-keepalive
	 no atm enable-ilmi-trap
	!
	interface ATM3/0.1 multipoint
	 ip address 10.1.1.2 255.255.255.0
	 no atm enable-ilmi-trap
	 pvc 0/50
	  protocol ip 10.1.1.1
	  encapsulation aal5snap
		interface ATM0/1/0
		 no ip address
		 atm clock INTERNAL
		 no atm enable-ilmi-trap
		!
		interface ATM0/1/0.50 point-to-point
		 no atm enable-ilmi-trap
		 pvc 0/50 l2transport
		  encapsulation aal5
		 !
		!
		interface ATM0/1/1
		 no ip address
		 atm clock INTERNAL
		 no atm enable-ilmi-trap
		!
		interface ATM0/1/1.100 point-to-point
		 no atm enable-ilmi-trap
		 pvc 0/100 l2transport
		  encapsulation aal5
		
		connect con_atm ATM0/1/1 0/100 ATM0/1/0 0/50

ATM VP-to-VP Local Switching (Different Port) Example

The following example shows the configuration of ATM VP-to-VP local switching:

CE1 router	CE2 router	PE router

`interface ATM1/0`
` no ip address`
` atm clock INTERNAL`
` no atm enable-ilmi-trap`
`!`
`interface ATM1/0.1 point-to-point`
` ip address 10.1.1.1 255.255.255.0`
` no atm enable-ilmi-trap`
` pvc 100/100`
`  encapsulation aal5snap`
	`interface ATM3/0`
	` no ip address`
	` atm clock INTERNAL`
	` no atm ilmi-keepalive`
	` no atm enable-ilmi-trap`
	`!`
	`interface ATM3/0.1 point-to-point`
	` ip address 10.1.1.2 255.255.255.0`
	` no atm enable-ilmi-trap`
	` pvc 100/100`
	`  encapsulation aal5snap`
		`interface ATM0/1/0`
		` no ip address`
		` atm clock INTERNAL`
		` no atm ilmi-keepalive`
		` no atm enable-ilmi-trap`
		`!`
		`interface ATM0/1/0.50 multipoint`
		` atm pvp 100 l2transport`
		` no atm enable-ilmi-trap`
		`!`
		`interface ATM0/1/1`
		` no ip address`
		` atm clock INTERNAL`
		` no atm ilmi-keepalive`
		` no atm enable-ilmi-trap`
		`!`
		`interface ATM0/1/1.100 multipoint`
		` atm pvp 100 l2transport`
		` no atm enable-ilmi-trap`
		`connect atm_con ATM0/1/1 100 ATM0/1/0 100`

Example: Configuring HDLC-to-Ethernet Interworking: Controller Slot on HDLC Devices

The following example shows how to configure the serial controller and interface on HDLC devices:

<table>
<tr><th>HDLC CE device</th><th>HDLC PE device</th></tr>
<tr><td><pre>
enable
 configure terminal
  controller E1 2/0
   channel-group 0 timeslots
1
   no shutdown
!
interface serial 2/0:0
 no shutdown
end
</pre></td><td><pre>
enable
 configure terminal
  controller E1 0/1/0
   channel-group 0 timeslots
1
   no shutdown
!
interface serial 0/1/0:0
 no shutdown
end
</pre></td></tr>
</table>

Example: Configuring HDLC-to-Ethernet Bridged Interworking on HDLC Devices

The following example shows how to configure HDLC-to-Ethernet bridged interworking on HDLC devices:

<table>
<tr><th>HDLC CE device</th><th>HDLC PE device</th></tr>
<tr><td><pre>
enable
 configure terminal
  bridge irb
  bridge 1 protocol ieee
  bridge 1 route ip
!
interface BVI1
 ip address 192.0.2.1 255.255.255.0

 no shutdown
!
interface serial 2/0:0
 encapsulation hdlc
 bridge-group 1
 no shutdown
end
</pre></td><td><pre>
enable
 configure terminal
  pseudowire-class pw-iw-eth
  encapsulation mpls
  interworking Ethernet
!
interface serial 0/1/0:0
 encapsulation hdlc
 no ip address
 xconnect 203.0.113.10 100 pw-class pw-iw-eth

 no shutdown
end
</pre></td></tr>
</table>

Example: Configuring HDLC-to-Ethernet Bridged Interworking on HDLC Devices Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

The following example shows how to configure HDLC-to-Ethernet bridged interworking on HDLC devices using the commands associated with the L2VPN protocol-based CLIs feature:

HDLC CE device	HDLC PE device

enable
 configure terminal
  bridge irb
  bridge 1 protocol ieee
  bridge 1 route ip
!
interface BVI1
 ip address 192.0.2.1 255.255.255.0
 	
 no shutdown
!
interface serial 2/0:0
 encapsulation hdlc
 bridge-group 1
 no shutdown
end
	enable
	 configure terminal
	  interface serial 0/1/0:0
	  encapsulation hdlc
	  no ip address
	  no shutdown
	!
	interface pseudowire 101
	 encapsulation mpls
	 neighbor 203.0.113.10 100
	 signaling protocol ldp
	 no shutdown
	!
	l2vpn xconnect context hdlc
	
	 interworking ethernet
	 member Serial 0/1/0:0
	 member pseudowire 101
	 no shutdown
	end

Example: Configuring HDLC-to-Ethernet Bridged Interworking on Ethernet Devices

The following example shows how to configure HDLC-to-Ethernet bridged interworking on Ethernet devices:

Ethernet CE device	Ethernet PE device

enable
 configure terminal
  interface GigabitEthernet0/1
   ip address 198.51.100.19 255.255.255.0
 	
   ip irdp
   ip irdp maxadvertinterval 4
   no shutdown
end
	enable
	 configure terminal
	 pseudowire-class pw-iw-eth
	  encapsulation mpls
	  interworking Ethernet
	!
	interface GigabitEthernet 1/0/0
	 no ip address
	 xconnect 203.0.113.20 100 pseudowire-class
	pw-iw-eth
	  no shutdown
	end

Example: Configuring HDLC-to-Ethernet Bridged Interworking on Ethernet Devices Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

The following example shows how to configure HDLC-to-Ethernet bridged interworking on Ethernet devices using the commands associated with the L2VPN protocol-based CLIs feature:

Ethernet CE device	Ethernet PE device

<pre>enable
configure terminal
interface GigabitEthernet 0/1
 ip address 198.51.100.19 255.255.255.0
 	
 ip irdp
 ip irdp maxadvertinterval 4
 no shutdown
end</pre>
	<pre>enable
	configure terminal
	interface GigabitEthernet 1/0/0
	
	 no ip address
	 no shutdown
	!
	interface pseudowire 101
	 encapsulation mpls
	 neighbor 203.0.113.20 100
	 signaling protocol ldp
	 no shutdown
	!
	l2vpn xconnect context eth
	 interworking ethernet
	 member GigabitEthernet 1/0/0
	 member pseudowire101
	 no shutdown
	end</pre>

Example: Configuring HDLC-to-VLAN Bridged Interworking (Port Mode) on Ethernet Devices

The following example shows how to configure HDLC-to-VLAN bridged interworking (port mode) on Ethernet devices:

Ethernet CE device	Ethernet PE device

<pre>enable
 configure terminal
  interface GigabitEthernet 0/1
   no ip address
   no shutdown
!
interface GigabitEthernet 0/1.10
 encapsulation dot1q 10
 ip address 198.51.100.19 255.255.255.0
 	
 ip irdp
 ip irdp maxadvertinterval 4
 no shutdown
end</pre>
	<pre>enable
	 configure terminal
	  pseudowire-class pw-iw-eth
	   encapsulation mpls
	   interworking Ethernet
	!
	interface GigabitEthernet 1/0/0
	 no ip address
	 no shutdown
	!
	interface GigabitEthernet 1/0/0.10
	 encapsulation dot1Q 10
	 no ip address
	!
	 xconnect 203.0.113.20 100 pseudowire-class
	pw-iw-eth
	  no shutdown
	end</pre>

Example: Configuring HDLC-to-VLAN Bridged Interworking on Ethernet Devices Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

The following example shows how to configure HDLC-to-VLAN bridged interworking on Ethernet devices using the commands associated with the L2VPN protocol-based CLIs feature:

Ethernet CE device	Ethernet PE device

enable
 configure terminal
  interface GigabitEthernet 0/1
   no ip address
   no shutdown
!
interface GigabitEthernet 0/1.10
 encapsulation dot1q 10
 ip address 198.51.100.19 255.255.255.0
 ip irdp
 ip irdp maxadvertinterval 4
 no shutdown
end
	enable
	 configure terminal
	  interface GigabitEthernet 1/0/0
	   no ip address
	   no shutdown
	!
	interface GigabitEthernet 1/0/0.10
	 encapsulation dot1q 10
	 no ip addres
	 no shutdown
	!
	interface pseudowire 101
	 encapsulation mpls
	 neighbor 203.0.113.20 100
	 signaling protocol ldp
	 no shutdown
	!
	l2vpn xconnect context vlan
	 interworking ethernet
	 member GigabitEthernet 1/0/0.10
	 member pseudowire 101
	 no shutdown
	end

Example: Configuring HDLC-to-VLAN Bridged Interworking (dot1q Mode) Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

The following example shows how to configure HDLC-to-VLAN bridged interworking (dot1q mode) using the commands associated with the L2VPN protocol-based CLIs feature:

HDLC PE device	Ethernet PE device

<pre>enable
 configure terminal
  template type pseudowire hdlc-vlan1
   encapsulation mpls
!
interface pseudowire 107
 source template type pseudowire hdlc-vlan1
 	
  encapsulation mpls
  neighbor 203.0.113.10 107
  signaling protocol ldp
  no shutdown
!
l2vpn xconnect context hdlc-vlan1-con
 interworking ethernet
 member Serial 0/2/0:3
 member pseudowire 107
 no shutdown
end</pre>
	<pre>enable
	 configure terminal
	  interface FastEthernet 0/0/0.16
	   encapsulation dot1q 16
	   no ip addres
	   no shutdown
	!
	template type pseudowire hdlc-vlan1
	 encapsulation mpls
	!
	interface pseudowire 107
	 source template type pseudowire hdlc-vlan1
	
	  encapsulation mpls
	  neighbor 203.0.113.20 107
	  signaling protocol ldp
	  no shutdown
	!
	l2vpn xconnect context hdlc-vlan1-con
	 interworking ethernet
	 member FastEthernet 0/0/0.16
	 member pseudowire 107
	 no shutdown
	end</pre>

Example: Configuring HDLC-to-VLAN Bridged Interworking (QinQ Mode) on Ethernet Devices

The following example shows how to configure HDLC-to-VLAN bridged interworking (QinQ mode) on Ethernet devices:

Ethernet CE device	Ethernet PE device

<pre>enable
 configure terminal
  interface GigabitEthernet 0/1
  no ip address
  no shutdown
!
interface GigabitEthernet 0/1.10
 encapsulation dot1q 10 second-dot1q 20
 	
 ip address 198.51.100.19 255.255.255.0
 	
 ip irdp
 ip irdp maxadvertinterval 4
 no shutdown
end</pre>
	<pre>enable
	 configure terminal
	  pseudowire-class pw-iw-eth
	  encapsulation mpls
	  interworking Ethernet
	!
	interface GigabitEthernet 1/0/0
	 no ip address
	 no shutdown
	!
	interface GigabitEthernet 1/0/0.10
	 encapsulation dot1Q 10 second-dot1q 20
	 no ip address
	 xconnect 203.0.113.20 100 pseudowire-class
	pw-iw-eth
	  no shutdown
	end</pre>

Example: Configuring HDLC-to-VLAN Bridged Interworking (QinQ Mode) on Ethernet Devices Using the Commands Associated with the L2VPN Protocol-Based CLIs Feature

The following example shows how to configure HDLC-to-VLAN bridged interworking (QinQ mode) on Ethernet devices using the commands associated with the L2VPN protocol-based CLIs feature:

Ethernet CE device	Ethernet PE device

`enable`
` configure terminal`
`  interface GigabitEthernet 0/1`
`  no ip address`
`  no shutdown`
`!`
`interface GigabitEthernet 0/1.10`
` encapsulation dot1q 10 second-dot1q 20`
 	
` ip address 198.51.100.19 255.255.255.0`
 	
` ip irdp`
` ip irdp maxadvertinterval 4`
` no shutdown`
`end`
	`enable`
	` configure terminal`
	`  interface GigabitEthernet 1/0/0`
	`  no ip address`
	`  no shutdown`
	`!`
	`interface GigabitEthernet 1/0/0.10`
	` encapsulation dot1q 10 second-dot1q 20`
	
	` no ip address`
	` no shutdown`
	`!`
	`interface pseudowire 101`
	` encapsulation mpls`
	` neighbor 203.0.113.20 100`
	` signaling protocol ldp`
	` no shutdown`
	`!`
	`l2vpn xconnect context qinq`
	` interworking ethernet`
	` member GigabitEthernet 1/0/0.10`
	` member pseudowire 101`
	` no shutdown`
	`end`

Additional References for L2VPN Interworking

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference
Any Transport over MPLS	Any Transport over MPLS

Standards and RFCs

Standard/RFC	Title
draft-ietf-l2tpext-l2tp-base-03.txt	*Layer Two Tunneling Protocol (Version 3) 'L2TPv3'*

Standard/RFC	Title
draft-martini-l2circuit-trans-mpls-09.txt	*Transport of Layer 2 Frames Over MPLS*
draft-ietf-pwe3-frame-relay-03.txt.	*Encapsulation Methods for Transport of Frame Relay over MPLS Networks*
draft-martini-l2circuit-encap-mpls-04.txt.	*Encapsulation Methods for Transport of Layer 2 Frames Over IP and MPLS Networks*
draft-ietf-pwe3-ethernet-encap-08.txt.	*Encapsulation Methods for Transport of Ethernet over MPLS Networks*
draft-ietf-pwe3-hdlc-ppp-encap-mpls-03.txt.	*Encapsulation Methods for Transport of PPP/HDLC over MPLS Networks*
draft-ietf-ppvpn-l2vpn-00.txt.	*An Architecture for L2VPNs*
RFC 4618	Encapsulation Methods for Transport of PPP/High-Level Data Link Control (HDLC) over MPLS Networks

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register on Cisco.com.	http://www.cisco.com/techsupport

Feature Information for L2VPN Interworking

Table 13: Feature Information for L2VPN Interworking

Feature Name	Releases	Feature Information
L2VPN Interworking	Cisco IOS XE Release 2.4 Cisco IOS XE Release 3.3S	This feature allows disparate ACs to be connected. An interworking function facilitates the translation between the different Layer 2 encapsulations. The following commands were introduced or modified: **debug frame-relay pseudowire**, **debug ssm**, **interworking**, **mtu**, **pseudowire-class**, **show l2tun session**, **show l2tun tunnel** , **show mpls l2transport vc**, **show platform**.
L2VPN Interworking: Ethernet to VLAN Interworking	Cisco IOS XE Release 2.4	This feature allows interworking by stripping the VLAN tags and sending them as untagged frames on the remote end.
L2VPN Interworking: Ethernet VLAN to Frame Relay	Cisco IOS XE Release 3.3S	This feature allows interworking of Ethernet VLANs with Frame Relay DLCIs. The following command was modified: **interworking**
L2VPN Interworking: Ethernet VLAN to PPP	Cisco IOS XE Release 3.3S	The L2VPN interworking - Ethernet VLAN-to-PPP feature allows disparate ACs to be connected. An interworking function facilitates the translation between the following Layer 2 encapsulations.
L2VPN Interworking: Frame Relay to ATM (Bridged Mode)	Cisco IOS XE Release 3.6S	This feature allows Frame Relay to ATM Interworking using bridged and routed mode encapsulation.
L2VPN Interworking: HDLC to Ethernet Interworking	Cisco IOS XE Release 3.13S	High-Level Data Link Control (HDLC) and Ethernet are two independent data link layer transport protocols that utilize the Any Transport over MPLS (AToM) framework to communicate with each other. The interworking function enables translation between two heterogeneous Layer 2 encapsulations over a Multiprotocol Label Switching (MPLS) backbone. In Cisco IOS XE Release 3.13S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. This feature introduced no new or modified commands.

CHAPTER 5

L2VPN Pseudowire Preferential Forwarding

The L2VPN: Pseudowire Preferential Forwarding feature allows you to configure the pseudowires so that you can use **ping** and **show** commands to find status information for the pseudowires before, during, and after a switchover.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for L2VPN—Pseudowire Preferential Forwarding

- Before configuring the L2VPN: Pseudowire Preferential Forwarding feature, you should understand the concepts in the following documents:
  - Preferential Forwarding Status Bit Definition (draft-ietf-pwe3-redundancy-bit-xx.txt)
  - *MPLS Pseudowire Status Signaling*
  - *L2VPN Pseudowire Redundancy*
  - *NSF/SSO--Any Transport over MPLS and AToM Graceful Restart*
  - *MPLS LSP Ping/Traceroute for LDP/TE, and LSP Ping for VCCV*
- The PE routers must be configured with the following features:

    - *L2VPN Pseudowire Redundancy*
    - *NSF/SSO--Any Transport over MPLS and AToM Graceful Restart*

- The L2VPN: Pseudowire Preferential Forwarding feature requires that the following mechanisms be in place to enable you to detect a failure in the network:
    - *Label switched paths (LSPs) Ping/Traceroute and Any Transport over MPLS Virtual Circuit Connection Verification (AToM VCCV)*
    - *Local Management Interface (LMI)*
    - *Operation, Administration, and Maintenance (OAM)*

Guidelines and Limitations for L2VPN--Pseudowire Preferential Forwarding

- Only ATM attachment circuits are supported.
- The following features are not supported:
    - Port mode cell relay
    - Any Transport over MPLS: AAL5 over MPLS
    - VC cell packing
    - OAM emulation
    - ILMI/PVC-D
    - Permanent virtual circuit (PVC) Range
    - L2TPv3 Pseudowire Redundancy
    - Local switching
    - Multiple backup pseudowires
    - Static pseudowires

Information About L2VPN--Pseudowire Preferential Forwarding

Overview of L2VPN--Pseudowire Preferential Forwarding

The L2VPN: Pseudowire Preferential Forwarding feature allows you to configure pseudowires so that you can use **ping** , **traceroute** , and **show** commands to find status information before, during, and after a switchover. The implementation of this feature is based on *Preferential Forwarding Status Bit Definition* (draft-ietf-pwe3-redundancy-bit-xx.txt). The L2VPN: Pseudowire Preferential Forwarding feature provides the following enhancements for displaying information about the pseudowires:

- You can issue **ping mpls** commands on the backup pseudowires.
- You can display status of the pseudowires before, during, and after a switchover using the **show xconnect** and **show mpls l2transport vc** commands.

Note In a single-segment pseudowire, the PE routers at each end of the pseudowire serve as the termination points. In multisegment pseudowires, the terminating PE routers serve as the termination points.

Overview of L2VPN—Pseudowire Preferential Forwarding using the commands associated with the L2VPN Protocol-Based CLIs feature

The L2VPN: Pseudowire Preferential Forwarding feature allows you to configure pseudowires so that you can use **ping**, **traceroute**, and **show** commands to find status information before, during, and after a switchover. The implementation of this feature is based on *Preferential Forwarding Status Bit Definition* (draft-ietf-pwe3-redundancy-bit-xx.txt). The L2VPN: Pseudowire Preferential Forwarding feature provides the following enhancements for displaying information about the pseudowires:

- You can issue **ping mpls** commands on the backup pseudowires.
- You can display status of the pseudowires before, during, and after a switchover using the **show l2vpn service** and **show l2vpn atom vc** commands.

Note In a single-segment pseudowire, the PE routers at each end of the pseudowire serve as the termination points. In multisegment pseudowires, the terminating PE routers serve as the termination points.

How to Configure L2VPN--Pseudowire Preferential Forwarding

Configuring the Pseudowire Connection Between PE Routers

You set up a connection called a pseudowire between the routers to transmit Layer 2 frames between PE routers.

As part of the pseudowire configuration, issue the **status redundancy master** command to make it the master. This enables the L2VPN: Pseudowire Preferential Forwarding feature to display the status of the active and backup pseudowires. By default, the PE router is in slave mode.

Note One pseudowire must be the master, and the other must be the slave. You cannot configure both pseudowires as master or slave.

Note You must specify the **encapsulation mpls** command as part of the pseudowire class in order for the AToM VCs to work properly. If you omit the **encapsulation mpls** command, you receive the following error: % Incomplete command.

Before you begin

The PE routers must be configured for the L2VPN Pseudowire Redundancy and NSF/SSO--Any Transport over MPLS and AToM Graceful Restart features. See the following documents for configuration instructions.

- *L2VPN Pseudowire Redundancy*
- *NSF/SSO--Any Transport over MPLS and AToM Graceful Restart*

SUMMARY STEPS

1. **configure terminal**
2. **pseudowire-class name**
3. **encapsulation mpls**
4. **status redundancy {master| slave}**
5. **interworking {ethernet | ip}**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**pseudowire-class name** **Example:** `switch(config)# pseudowire-class atom`	Establishes a pseudowire class with a name that you specify, and enters pseudowire class configuration mode.
Step 3	**encapsulation mpls** **Example:** `switch(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is mpls.
Step 4	**status redundancy {master| slave}** **Example:** `switch(config-pw)# status redundancy master`	Configures the pseudowire as the master or slave. This enables the L2VPN: Pseudowire Preferential Forwarding feature to display the status of the active and backup pseudowires. • By default, the PE router is in slave mode. **Note** One pseudowire must be the master, and the other must be the slave. You cannot configure both pseudowires as master or slave.
Step 5	**interworking {ethernet | ip}** **Example:** `switch(config-pw)# interworking ip`	(Optional) Enables the translation between the different Layer 2 encapsulations.

Configuring the Pseudowire Connection Between PE Routers

You set up a connection called a pseudowire between the routers to transmit Layer 2 frames between PE routers.

As part of the pseudowire configuration, issue the **status redundancy master** command to make it the master. This enables the L2VPN: Pseudowire Preferential Forwarding feature to display the status of the active and backup pseudowires. By default, the PE router is in slave mode.

Note One pseudowire must be the master, and the other must be the slave. You cannot configure both pseudowires as master or slave.

Note You must specify the **encapsulation mpls** command as part of the pseudowire class in order for the AToM VCs to work properly. If you omit the **encapsulation mpls** command, you receive the following error: % Incomplete command.

Before you begin

The PE routers must be configured for the L2VPN Pseudowire Redundancy and NSF/SSO--Any Transport over MPLS and AToM Graceful Restart features. See the following documents for configuration instructions.

- *L2VPN Pseudowire Redundancy*
- *NSF/SSO--Any Transport over MPLS and AToM Graceful Restart*

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *number*
4. **encapsulation mpls**
5. **neighbor** *peer-address vcid-value*
6. **status redundancy** {**master**| **slave**}
7. **interworking** {**ethernet** | **ip**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 1`	Establishes an interface pseudowire with a value that you specify, and enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation. • For AToM, the encapsulation type is mpls.
Step 5	**neighbor** *peer-address* *vcid-value* **Example:** `Router(config-pw)#  neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 6	**status redundancy** {**master**| **slave**} **Example:** `Device(config-pw)# status redundancy master`	Configures the pseudowire as the master or slave. This enables the L2VPN: Pseudowire Preferential Forwarding feature to display the status of the active and backup pseudowires. • By default, the PE router is in slave mode. **Note** One pseudowire must be the master, and the other must be the slave. You cannot configure both pseudowires as master or slave.
Step 7	**interworking** {**ethernet** | **ip**} **Example:** `Device(config-pw)# interworking ip`	(Optional) Enables the translation between the different Layer 2 encapsulations.

Configuration Examples for L2VPN--Pseudowire Preferential Forwarding

Example: L2VPN--Pseudowire Preferential Forwarding Configuration

The following commands configure a PE router with the L2VPN: Pseudowire Preferential Forwarding feature:

```
mpls ldp graceful-restart
mpls ip
mpls label protocol ldp
mpls ldp router-id Loopback0 force
mpls ldp advertise-labels
```

```
!
pseudowire-class mpls
 encapsulation mpls
 status redundancy master
interface ATM0/2/0.1 multipoint
 logging event subif-link-status
 atm pvp 50 l2transport
  xconnect 10.1.1.2 100 pw-class mpls
   backup peer 10.1.1.3 100 encap mpls
end
```

Example: L2VPN--Pseudowire Preferential Forwarding Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

The following commands configure a PE router with the L2VPN: Pseudowire Preferential Forwarding feature:

```
mpls ldp graceful-restart
mpls ip
mpls label protocol ldp
mpls ldp router-id Loopback0 force
mpls ldp advertise-labels
!
interface pseudowire1
 encapsulation mpls
 status redundancy master
 neighbor 10.0.0.1 123
interface ATM0/2/0.1 multipoint
 logging event subif-link-status
 atm pvp 50 l2transport
  interface pseudowire 100
  encapsulation mpls
  neighbor 10.1.1.2 100
!
l2vpn xconnect context A
 member pseudowire 100
 member atm 100
end
```

Example: Displaying the Status of the Pseudowires

The following examples show the status of the active and backup pseudowires before, during, and after a switchover.

The **show mpls l2transport vc** command on the active PE router displays the status of the pseudowires:

```
Router# show mpls l2transport vc

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.2         100        UP
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.3         100        STANDBY
```

The **show mpls l2transport vc** command on the backup PE router displays the status of the pseudowires. The active pseudowire on the backup PE router has the HOTSTANDBY status.

```
Router1-standby# show mpls l2transport vc
```

```
Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.2         100        HOTSTANDBY
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.3         100        DOWN
```

During a switchover, the status of the active and backup pseudowires changes:

```
Router# show mpls l2transport vc

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.2         100        RECOVERING
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.3         100        DOWN
```

After the switchover is complete, the recovering pseudowire shows a status of UP:

```
Router# show mpls l2transport vc

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.2         100        UP
AT0/2/0/0.1    ATM VPC CELL 50            10.1.1.3         100        STANDBY
```

The **show xconnect** command displays the standby (SB) state for the backup pseudowire, which is independent of the stateful switchover mode of the router:

```
Router# show xconnect all

Legend:    XC ST=Xconnect State  S1=Segment1 State  S2=Segment2 State
  UP=Up       DN=Down            AD=Admin Down      IA=Inactive
  SB=Standby  HS=Hot Standby     RV=Recovering      NH=No Hardware
XC ST        Segment 1                                  S1 Segment 2
            S2
------+--------------------------------+--+--------------------------------+---------
UP pri ac   AT1/1/0/0.1/1/1:220/220(ATM V  UP mpls 10.193.193.3:330            UP
IA sec ac   AT1/1/0/0.1/1/1:220/220(ATM V  UP mpls 10.193.193.3:331            SB
```

The **ping mpls** and **traceroute mpls** commands show that the dataplane is active on the backup pseudowire:

```
Router# ping mpls pseudowire 10.193.193.22 331

%Total number of MS-PW segments is less than segment number; Adjusting the segment number
to 1
Sending 5, 100-byte MPLS Echos to 10.193.193.22,
     timeout is 2 seconds, send interval is 0 msec:
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0
Type escape sequence to abort.
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

Router# traceroute mpls pseudowire 10.193.193.22 331 segment 1

Tracing MS-PW segments within range [1-1] peer address 10.193.193.22 and timeout 2 seconds
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
```

```
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0
Type escape sequence to abort.
! 1 10.193.33.22 4 ms [Labels: 23 Exp: 0]
    local 10.193.193.3 remote 10.193.193.22 vc id 331
```

Additional References

Related Documents

Related Topic	Document Title
Description of commands associated with MPLS and MPLS applications	*Cisco IOS Multiprotocol Label Switching Command Reference*
L2VPN Pseudowires	• *L2VPN Pseudowire Redundancy* • *MPLS Pseudowire Status Signaling*
NSF/SSO for L2VPNs	*NSF/SSO--Any Transport over MPLS and AToM Graceful Restart*
Ping and Traceroute for L2VPNs	*MPLS LSP Ping/Traceroute for LDP/TE, and LSP Ping for VCCV*

Standards

Standard	Title
draft-ietf-pwe3-redundancy-bit-xx.txt	Preferential Forwarding Status Bit Definition

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for L2VPN--Pseudowire Preferential Forwarding

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14: Feature Information for L2VPN: Pseudowire Preferential Forwarding

Feature Name	Releases	Feature Information
L2VPN: Pseudowire Preferential Forwarding	Cisco IOS XE Release 2.3	This feature allows you to configure the pseudowires so that you can use **ping** and **show** commands to find status information of the pseudowires before, during, and after a switchover. The following commands were introduced or modified: **show mpls l2transport vc**, **show xconnect**, **status redundancy**.

CHAPTER 6

L2VPN Multisegment Pseudowires

The L2VPN Multisegment Pseudowires feature enables you to configure two or more Layer 2 pseudowire segments that function as a single pseudowire. The L2VPN Multisegment Pseudowires feature span multiple cores or autonomous systems of the same or different carrier networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for L2VPN Multisegment Pseudowires

Before configuring this feature, see the following documents:

- Any Transport over MPLS
- *L2VPN Pseudowire Switching*
- MPLS LSP Ping/Traceroute for LDP/TE, and LSP Ping for VCCV
- Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP) (RFC 4447)

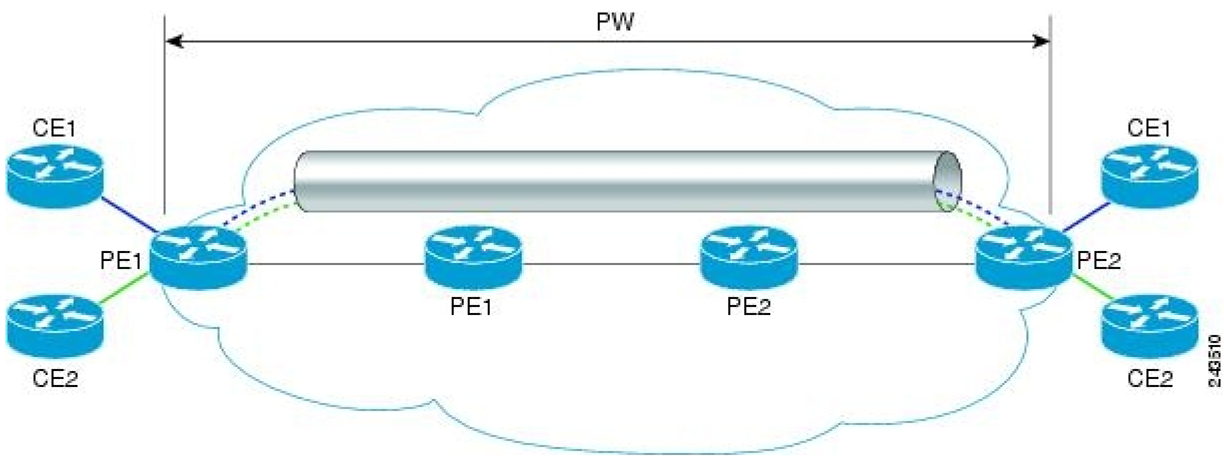
Restrictions for L2VPN Multisegment Pseudowires

- Only Mutliprotocol (MPLS) Layer 2 pseudowires are supported.
- Only manual configuration of the pseudowires (including S-PE and T-PE routers) is supported.
- The L2VPN Pseudowire Switching feature is supported for pseudowires advertised with FEC 128. FEC 129 is not supported.
- The S-PE router is limited to 1600 pseudowires.

Information About L2VPN Multisegment Pseudowires

L2VPN Pseudowire Defined

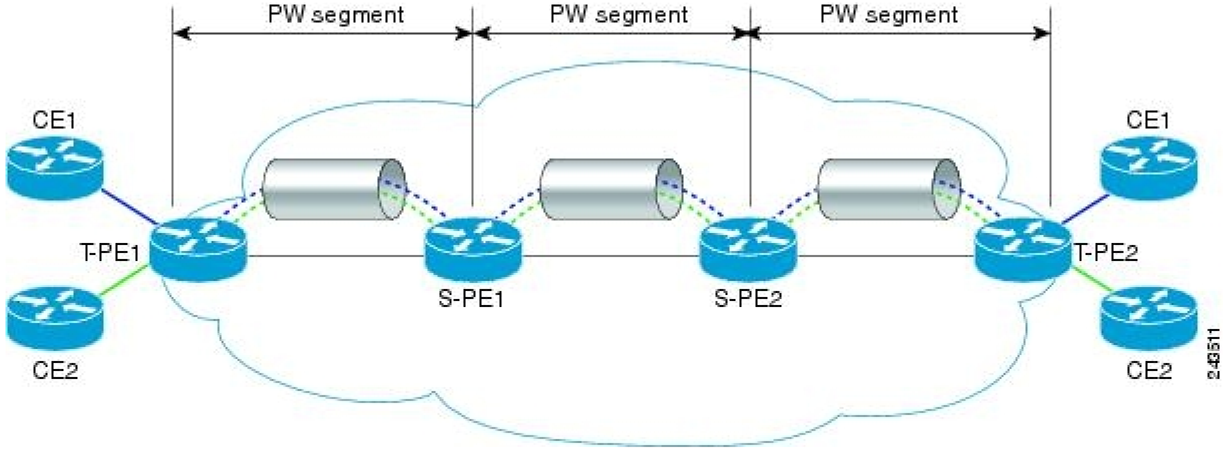
An L2VPN pseudowire (PW) is a tunnel established between two provider edge (PE) routers across the core carrying the Layer 2 payload encapsulated as MPLS data, as shown in the figure below. This helps carriers migrate from traditional Layer 2 networks such as Frame Relay and ATM to an MPLS core. In the L2VPN pseudowire shown in the figure, the PWs between two PE routers are located within the same autonomous system. Routers PE1 and PE2 are called terminating PE routers (T-PEs). Attachment circuits are bounded to the PW on these PE routers.



L2VPN Multisegment Pseudowire Defined

An L2VPN multisegment pseudowire (MS-PW) is a set of two or more PW segments that function as a single PW. It is also known as switched PW. MS-PWs span multiple cores or autonomous systems of the same or different carrier networks. A L2VPN MS-PW can include up to 254 PW segments.

The figure below is an example of a Multisegment Pseudowire topology.



The end routers are called terminating PE routers (T-PEs), and the switching routers are called S-PE routers. The S-PE router terminates the tunnels of the preceding and succeeding PW segments in an MS-PW. The S-PE router can switch the control and data planes of the preceding and succeeding PW segments of the MS-PW. An MS-PW is declared to be up when all the single-segment PWs are up. For more information, see the *L2VPN Pseudowire Switching* document.

How to Configure L2VPN Multisegment Pseudowires

Configuring L2VPN Multisegment Pseudowires

Perform the following steps on the S-PE routers to create L2VPN Multisegment Pseudowires.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. mpls label protocol ldp
4. **mpls ldp router-id** *interface* **force**
5. **pseudowire-class** name
6. **encapsulation mpls**
7. **switching tlv**
8. **exit**
9. **l2 vfi** *name* **point-to-point**
10. **description** *string*
11. **neighbor** *ip-address vcid {* **encapsulation mpls** **pw-class** *pw-class-name}*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	mpls label protocol ldp **Example:** `Router(config)# mpls label protocol ldp`	Configures the use of Label Distribution Protocol (LDP) on all interfaces.
Step 4	**mpls ldp router-id** *interface* **force** **Example:** `Router(config)# mpls ldp router-id loopback0 force`	Specifies the preferred interface for determining the LDP router ID.
Step 5	**pseudowire-class** name **Example:** `Router(config)# pseudowire-class atom`	Establishes a pseudowire class with a name that you specify, and enters pseudowire class configuration mode.
Step 6	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation. • For MPLS L2VPNs, the encapsulation type is **mpls**.
Step 7	**switching tlv** **Example:** `Router(config-pw-class)# switching tlv`	(Optional) Enables the advertisement of the switching point type-length variable (TLV) in the label binding. • This command is enabled by default.
Step 8	**exit** **Example:** `Router(config-pw-class)# exit`	Exits pseudowire class configuration mode.
Step 9	**l2 vfi** *name* **point-to-point** **Example:** `Router(config)# l2 vfi atomtunnel point-to-point`	Creates a point-to-point Layer 2 virtual forwarding interface (VFI) and enters VFI configuration mode.
Step 10	**description** *string* **Example:** `Router(config-vfi)# description segment1`	Provides a description of the switching provider edge router for a multisegment pseudowire.

	Command or Action	Purpose
Step 11	**neighbor** *ip-address vcid {* **encapsulation mpls** **pw-class** *pw-class-name}* **Example:** `Router(config-vfi)# neighbor 10.0.0.1 100 pw-class mpls`	Sets up an emulated VC. • Specify the IP address and the VC ID of the peer router. Also specify the pseudowire class to use for the emulated VC. **Note** Only two **neighbor**commands are allowed for each **l2 vfi point-to-point** command.

Configuring L2VPN Multisegment Pseudowires using the commands associated with the L2VPN Protocol-Based CLIs feature

Perform this task on the S-PE routers to create L2VPN multisegment pseudowires.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol ldp**
4. **mpls ldp router-id** *interface* **force**
5. **interface pseudowire** *number*
6. **encapsulation mpls**
7. **switching tlv**
8. **neighbor** *peer-address vcid-value*
9. **exit**
10. **l2vpn xconnect context** *context-name*
11. **description** *string*
12. **member** *ip-address vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol ldp** **Example:**	Configures the use of Label Distribution Protocol (LDP) on all interfaces.

	Command or Action	Purpose
	`Device(config)# mpls label protocol ldp`	
Step 4	**mpls ldp router-id** *interface* **force** **Example:** `Device(config)# mpls ldp router-id loopback0 force`	Specifies the preferred interface for determining the LDP router ID.
Step 5	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 1`	Establishes an interface pseudowire with a value that you specify, and enters pseudowire configuration mode.
Step 6	**encapsulation mpls** **Example:** `Device(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation. • For MPLS L2VPNs, the encapsulation type is **mpls**.
Step 7	**switching tlv** **Example:** `Device(config-pw)# switching tlv`	(Optional) Enables the advertisement of the switching point type-length variable (TLV) in the label binding. • This command is enabled by default.
Step 8	**neighbor** *peer-address vcid-value* **Example:** `Router(config-pw)#  neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 9	**exit** **Example:** `Device(config-pw)# exit`	Exits pseudowire configuration mode.
Step 10	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 11	**description** *string* **Example:** `Device(config-xconnect)# description segment1`	Provides a description of the switching provider edge router for a multisegment pseudowire.
Step 12	**member** *ip-address vcid* **encapsulation mpls** **Example:** `Device(config-xconnect)# member 10.10.10.10 1 encapsulation mpls`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection. **Note** Only two **member**commands are allowed for each **l2vpn xconnect context** command.

Displaying Information About the L2VPN Multisegment Pseudowires

SUMMARY STEPS

1. **show mpls l2transport binding**
2. **show mpls l2transport vc detail**

DETAILED STEPS

Step 1 **show mpls l2transport binding**

Use the **show mpls l2transport binding** command to display information about the pseudowire switching point, as shown in bold in the output. (In the following examples PE1 and PE4 are the T-PE routers.)

Example:

```
Router# show mpls l2transport binding

  Destination Address: 10.1.1.1,  VC ID: 102
    Local Label:  17
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2], TTL [3]
              CV Type: LSPV [2]
    Remote Label: 16
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2], TTL [3]
              CV Type: LSPV [2]
        PW Switching Point:
             Vcid   local IP addr      remote IP addr      Description
             101    10.11.11.11        10.20.20.20         PW Switching Point PE3
             100    10.20.20.20        10.11.11.11         PW Switching Point PE2
```

Step 2 **show mpls l2transport vc detail**

Use the **show mpls l2transport vc detail** command to display status of the pseudowire switching point. In the following example, the output (shown in bold) displays the segment that is the source of the fault of the multisegment pseudowire:

Example:

```
Router# show mpls l2transport vc detail
Local interface: Se3/0/0 up, line protocol up, HDLC up
  Destination address: 12.1.1.1, VC ID: 100, VC status: down
    Output interface: Se2/0, imposed label stack {23}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:03:02, last status change time: 00:01:41
  Signaling protocol: LDP, peer 10.1.1.1:0 up
    Targeted Hello: 10.1.1.4(LDP Id) -> 10.1.1.1, LDP is UP
    Status TLV support (local/remote)   : enabled/supported
      LDP route watch                   : enabled
      Label/status state machine        : established, LruRrd
      Last local dataplane   status rcvd: No fault
      Last local SSS circuit status rcvd: No fault
      Last local SSS circuit status sent: DOWN(PW-tx-fault)
      Last local  LDP TLV    status sent: No fault
```

```
      Last remote LDP TLV    status rcvd: DOWN(PW-tx-fault)
       PW Switching Point:
       Fault type  Vcid   local IP addr   remote IP addr   Description
       PW-tx-fault  101   10.1.1.1        10.1.1.1         S-PE2
      Last remote LDP ADJ    status rcvd: No fault
    MPLS VC labels: local 19, remote 23
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 16, send 27
    byte totals:   receive 2506, send 3098
    packet drops:  receive 0, seq error 0, send 0
```

Displaying Information About the L2VPN Multisegment Pseudowires using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **show l2vpn atom binding**
2. **show l2vpn atom vc detail**

DETAILED STEPS

Step 1 **show l2vpn atom binding**

Use the **show l2vpn atom binding** command to display information about the pseudowire switching point, as shown in bold in the output. (In the following examples PE1 and PE4 are the T-PE routers.)

Example:

```
Device# show l2vpn atom binding

  Destination Address: 10.1.1.1,  VC ID: 102
    Local Label:  17
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2], TTL [3]
              CV Type: LSPV [2]
    Remote Label: 16
        Cbit: 1,    VC Type: FastEthernet,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2], TTL [3]
              CV Type: LSPV [2]
        PW Switching Point:
             Vcid   local IP addr      remote IP addr      Description
             101    10.11.11.11        10.20.20.20         PW Switching Point PE3
             100    10.20.20.20        10.11.11.11         PW Switching Point PE2
```

Step 2 **show l2vpn atom vc detail**

Use the **show l2vpn atom vc detail** command to display status of the pseudowire switching point. In the following example, the output (shown in bold) displays the segment that is the source of the fault of the multisegment pseudowire:

Example:

```
Device# show l2vpn atom vc detail
Local interface: Se3/0/0 up, line protocol up, HDLC up
  Destination address: 12.1.1.1, VC ID: 100, VC status: down
    Output interface: Se2/0, imposed label stack {23}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:03:02, last status change time: 00:01:41
  Signaling protocol: LDP, peer 10.1.1.1:0 up
    Targeted Hello: 10.1.1.4(LDP Id) -> 10.1.1.1, LDP is UP
    Status TLV support (local/remote)   : enabled/supported
      LDP route watch                   : enabled
      Label/status state machine        : established, LruRrd
      Last local dataplane   status rcvd: No fault
      Last local SSS circuit status rcvd: No fault
      Last local SSS circuit status sent: DOWN(PW-tx-fault)
      Last local  LDP TLV    status sent: No fault
      Last remote LDP TLV    status rcvd: DOWN(PW-tx-fault)
       PW Switching Point:
       Fault type  Vcid   local IP addr   remote IP addr   Description
       PW-tx-fault  101   10.1.1.1        10.1.1.1         S-PE2
      Last remote LDP ADJ    status rcvd: No fault
    MPLS VC labels: local 19, remote 23
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 16, send 27
    byte totals:   receive 2506, send 3098
    packet drops:  receive 0, seq error 0, send 0
```

Performing ping mpls and trace mpls Operations on the L2VPN Multisegment Pseudowires

You can use the **ping mpls** and **trace mpls**commands to verify that all the segments of the MPLS multisegment pseudowire are operating.

You can use the **ping mpls** command to verify connectivity at the following pseudowire points:

- From one end of the pseudowire to the other
- From one of the pseudowires to a specific segment
- The segment between two adjacent S-PE routers

You can use the **trace mpls**command to verify connectivity at the following pseudowire points:

- From one end of the pseudowire to the other
- From one of the pseudowires to a specific segment
- The segment between two adjacent S-PE routers
- A range of segments

SUMMARY STEPS

1. **ping mpls pseudowire** *destination-address* *vc-id* [**segment** *segment-number*]
2. **trace mpls pseudowire** *destination-address vc-id* **segment** *segment-number* *segment-number*

DETAILED STEPS

Step 1 **ping mpls pseudowire** *destination-address* *vc-id* [**segment** *segment-number*]

Where:

- *destination-address* is the address of the S-PE router, which is the end of the segment from the direction of the source.
- *vc-id* is the VC ID of the segment from the source to the next PE router.
- **segment** *segment-number is optional and specifies the segment you want to ping.*

The following examples use the topology shown in the second figure above :

- To perform an end-to-end ping operation from T-PE1 to T-PE2, enter the following command:

ping mpls pseudowire *<addr-of-S-PE1> <vc-id between T-PE1 and S-PE1>*

- To perform a ping operation from T-PE1 to segment 2, enter the following command:

ping mpls pseudowire *<addr-of-S-PE1> <vc-id between T-PE1 and S-PE1>* **segment 2**

Example:

Step 2 **trace mpls pseudowire** *destination-address vc-id* **segment** *segment-number* *segment-number*

Where:

- *destination-address is the address of the next S-PE router from the original of the trace.*
- *vc-id* is the VC ID of the segment from which the **trace** command is issued.
- *segment-number* indicates the segment upon which the trace operation will act. If you enter two segment numbers, the traceroute operation will perform a trace on that range of routers.

The following examples use the topology shown in the second figure above :

- To perform a trace operation from T-PE1 to segment 2 of the multisegment pseudowire, enter the following command:

trace mpls pseudowire *<addr-of-S-PE1> <vc-id between T-PE1 and S-PE1>* **segment 2**

This example performs a trace from T-PE1 to S-PE2.

- To perform a trace operation on a range of segments, enter the following command. This example performs a trace from S-PE2 to T-PE2.

trace mpls pseudowire *<addr-of-S-PE1> <vc-id between T-PE1 and S-PE1>* **segment 2 4**

The following command performs a trace operation on S-PE router 10.10.10.9, on segment 1 and then on segment 2:

Example:

```
router# trace mpls pseudowire 10.10.10.9 220 segment 1
Tracing MS-PW segments within range [1-1] peer address 10.10.10.9 and timeout 2 seconds
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0
Type escape sequence to abort.
L 1 10.10.9.9 0 ms [Labels: 18 Exp: 0]
    local 10.10.10.22 remote 10.10.10.9 vc id 220
router# trace mpls pseudowire 10.10.10.9 220 segment 2
Tracing MS-PW segments within range [1-2] peer address 10.10.10.9 and timeout 2 seconds
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0
Type escape sequence to abort.
L 1 10.10.9.9 4 ms [Labels: 18 Exp: 0]
    local 10.10.10.22 remote 10.10.10.9 vc id 220
! 2 10.10.3.3 4 ms [Labels: 16 Exp: 0]
    local 10.10.10.9 remote 10.10.10.3 vc id 220
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Description of commands associated with MPLS and MPLS applications	*Cisco IOS Multiprotocol Label Switching Command Reference*
Layer 2 VPNS	• Any Transport over MPLS • *L2VPN Pseudowire Switching* • MPLS LSP Ping/Traceroute for LDP/TE, and LSP Ping for VCCV

Standards

Standard	Title
RFC 4777	Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for L2VPN Multisegment Pseudowires

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15: Feature Information for L2VPN Multisegment Pseudowires

Feature Name	Releases	Feature Information
MPLS OAM Support for Multisegment Pseudowires	Cisco IOS XE Release 2.3 Cisco IOS XE Release 3.5S	The L2VPN Multisegment Pseudowires feature enables you to configure two or more Layer 2 pseudowire segments that function as a single pseudowire. The L2VPN Multisegment Pseudowires feature span multiple cores or autonomous systems of the same or different carrier networks. In isco IOS XE Release 2.3, this feature was introduced and implemented on the Cisco ASR 1000 Series Routers. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 Router. The following commands were introduced or modified: **description** (l2 vfi), **ping mpls**, **show mpls l2transport binding**, **show mpls l2transport vc**, **switching tlv**, **trace mpls**.

CHAPTER 7

MPLS Quality of Service

The MPLS Quality of Service feature (formerly named as the MPLS CoS feature) enables you to provide differentiated services across an MPLS network. To satisfy a wide range of networking requirements, you can specify the class of service applicable to each transmitted IP packet. Different classes of service can be established for IP packets by setting the IP precedence bit in the header of each packet.

Prerequisites for MPLS Quality of Service

To use MPLS CoS to full advantage in your network, the following functionality must be supported:

- Multiprotocol Label Switching (MPLS)—MPLS is the standardized label switching protocol defined by the Internet Engineering Task Force (IETF).
- Cisco Express Forwarding—Cisco Express Forwarding is an advanced Layer 3 IP switching technology that optimizes performance and scalability in networks that handle large volumes of traffic and that exhibit dynamic traffic patterns.
- Asynchronous Transfer Mode (ATM)—ATM signaling support is required if you are using ATM interfaces in your network.

  If you are using only packet interfaces in your network, ATM functionality is not needed.
- QoS features:
  - Weighted fair queueing (WFQ)—Used on non-GSR platforms, WFQ is a dynamic scheduling method that allocates bandwidth fairly to all network traffic.

    WFQ applies priorities, or weights, to traffic to classify the traffic into flows and determine how much bandwidth to allow each flow. WFQ moves interactive traffic to the front of a queue to reduce response time and fairly shares the remaining bandwidth among high-bandwidth flows.
  - Weighted random early detection (WRED)—WRED is a congestion avoidance mechanism that extends RED functionality by allowing different RED parameters to be configured per IP precedence value.

IP precedence bits, contained in the type of service (ToS) octet in the IP packet header, are used to denote the relative importance or priority of an IP packet. WRED uses these IP precedence values to classify packets into different discard priorities or classes of service.

- Modified deficit round robin (MDRR)—Used only on GSR platforms, MDRR is a traffic class prioritization mechanism that incorporates emission priority as a facet of quality of service. MDRR is similar in function to WFQ on non-GSR platforms.

  In MDRR, IP traffic is mapped to different classes of service queues. A group of queues is assigned to each traffic destination. On the transmit side of the platform, a group of queues is defined on a per-interface basis; on the receive side of the platform, a group of queues is defined on a per-destination basis. IP packets are then mapped to these queues, based on their IP precedence value.

  These queues are serviced on a round-robin basis, except for a queue that has been defined to run in either of two ways: strict priority mode or alternate priority mode.

  In strict priority mode, the high priority queue is serviced whenever it is not empty; this ensures the lowest possible delay for high priority traffic. In this mode, however, the possibility exists that other traffic might not be serviced for long periods of time if the high priority queue is consuming most of the available bandwidth.

  In alternate priority mode, the traffic queues are serviced in turn, alternating between the high priority queue and the remaining queues.

- Committed access rate (CAR)—CAR is a QoS feature that limits the input or output transmission rate on an interface and classifies packets by setting the IP precedence value or the QoS group in the IP packet header.

Information About MPLS Quality of Service

MPLS Quality of Service Overview

MPLS CoS functionality enables network administrators to provide differentiated services across an MPLS network. Network administrators can satisfy a wide range of networking requirements by specifying the class of service applicable to each transmitted IP packet. Different classes of service can be established for IP packets by setting the IP precedence bit in the header of each packet.

MPLS CoS supports the following differentiated services in an MPLS network:

- Packet classification
- Congestion avoidance
- Congestion management

The table below describes the MPLS CoS services and functions.

Table 16: MPLS CoS Services and Functions

Service	CoS Function	Description
Packet classification	Committed access rate (CAR). Packets are classified at the edge of the network before labels are assigned.	CAR uses the type of service (ToS) bits in the IP header to classify packets according to input and output transmission rates. CAR is often configured on interfaces at the edge of a network in order to control traffic flowing into or out of the network. You can use CAR classification commands to classify or reclassify a packet.
Congestion avoidance	Weighted random early detection (WRED). Packet classes are differentiated based on drop probability.	WRED monitors network traffic to anticipate and prevent congestion at common network and internetwork bottlenecks. WRED can selectively discard lower priority traffic when an interface becomes congested; WRED can also provide differentiated performance characteristics for different classes of service.
Congestion management	Weighted fair queueing WFQ) for non-GSR platform. Packet classes are differentiated based on bandwidth requirements and finite delay characteristics. Modified deficit round robin (MDRR) for GSR platforms.	WFQ is an automated scheduling system that ensures fair bandwidth allocation to all network traffic. WFQ uses weights (priorities) to determine how much bandwidth each class of traffic is allocated. MDRR, similar in function to WFQ for non-GSR platforms, is a traffic prioritization scheme that maps IP traffic to different classes of service queues, based on the IP precedence value of each packet. The queues are then serviced on a round-robin basis.

MPLS CoS enables you to duplicate Cisco IP CoS (Layer 3) features as closely as possible in MPLS devices, including label edge switch routers (edge LSRs) and label switch routers (LSRs). MPLS CoS functions map nearly one-for-one to IP CoS functions on all types of interfaces.

Tag Switching and MPLS Terminology

The table below lists the existing legacy tag switching terms and the new, equivalent Multiprotocol Label Switching (MPLS) IETF terms used in this document and other related Cisco publications.

Table 17: Tag Switching Terms and Equivalent MPLS Terms

Old Designation	New Designation
Tag switching	Multiprotocol Label Switching
Tag (short for tag switching)	MPLS
Tag (item or packet)	Label
TDP (Tag Distribution Protocol)	LDP (Label Distribution Protocol). Cisco TDP and LDP (MPLS Label Distribution Protocol) closely parallel each other in function, but differ in detail, such as message formats and the commands required to configure the respective protocols and to monitor their operation
Tag switched	Label switched
TFIB (tag forwarding information base)	LFIB (label forwarding information base)
TSR (tag switching router)	LSR (label switching router)
TVC (tag VC, tag virtual circuit)	LVC (label VC, label virtual circuit)
TSP (tag switch path)	LSP (label switch path)

LSRs Used at the Edge of an MPLS Network

Label switching routers (LSRs) used at the edge of a Multiprotocol Label Switching (MPLS) network backbone are devices running MPLS software. The edge LSRs can be at the ingress or the egress of the network.

At the ingress of an MPLS network, devices process packets as follows:

1. IP packets enter the edge of the MPLS network at the edge LSR.
2. The edge LSR uses a classification mechanism such as the Modular Quality of Service Command-Line Interface (CLI) (MQC) to classify incoming IP packets and set the IP precedence value. Alternatively, IP packets can be received with the IP precedence value already set.
3. For each packet, the device performs a lookup on the IP address to determine the next-hop LSR.
4. The appropriate label is inserted into the packet, and the IP precedence bits are copied into the MPLS EXP bits in the label header.
5. The labeled packets are forwarded to the appropriate output interface for processing.
6. The packets are differentiated by class according to one of the following:
    - Drop probability—Weighted random early detection (WRED)
    - Bandwidth allocation and delay—Class-based weighted fair queueing (CBWFQ)

In either case, LSRs enforce the defined differentiation by continuing to employ WRED or CBWFQ on every ingress device.

At the egress of an MPLS network, devices process packets as follows:

1. MPLS-labeled packets enter the edge LSR from the MPLS network backbone.
2. The MPLS labels are removed and IP packets may be (re)classified.
3. For each packet, the device performs a lookup on the IP address to determine the packet's destination and forwards the packet to the destination interface for processing.
4. The packets are differentiated by the IP precedence values and treated appropriately, depending on the WRED or CBWFQ drop probability configuration.

LSRs Used at the Core of an MPLS Network

Label switching routers (LSRs) used at the core of a Multiprotocol Label Switching (MPLS) network are devices running MPLS software. These devices at the core of an MPLS network process packets as follows:

1. MPLS labeled packets coming from the edge devices or other core devices enter the core device.
2. A lookup is done at the core device to determine the next hop LSR.
3. An appropriate label is placed (swapped) on the packet and the MPLS EXP bits are copied.
4. The labeled packet is then forwarded to the output interface for processing.
5. The packets are differentiated by the MPLS EXP field marking and treated appropriately, depending on the weighted early random detection (WRED) and class-based weighted fair queueing (CBWFQ) configuration.

Benefits of MPLS CoS in IP Backbones

You realize the following benefits when you use MPLS CoS in a backbone consisting of IP devices running Multiprotocol Label Switching (MPLS):

- Efficient resource allocation—Weighted fair queueing (WFQ) is used to allocate bandwidth on a per-class and per-link basis, thereby guaranteeing a percentage of link bandwidth for network traffic.
- Packet differentiation—When IP packets traverse an MPLS network, packets are differentiated by mapping the IP precedence bits of the IP packets to the MPLS CoS bits in the MPLS EXP field. This mapping of bits enables the service provider to maintain end-to-end network guarantees and meet the provisions of customer service level agreements (SLAs).
- Future service enhancements—MPLS CoS provides building blocks for future service enhancements (such as virtual leased lines) by meeting bandwidth requirements.

How to Configure MPLS Quality of Service

Configuring WRED

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **random-detect**
5. **random-detect precedence** *min-threshold max-threshold mark-probability*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# gigabitethernet0/0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**random-detect** **Example:** `Device(config-if)# random-detect`	Configures the interface to use weighted random early detection/distributed weighted random early detection (WRED/DWRED).
Step 5	**random-detect precedence** *min-threshold max-threshold mark-probability* **Example:** `Device(config-if)# random-detect precedence 0 32 256 100`	Configures WRED/DWRED parameters per precedence value.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Verifying WRED

To verify weighted random early detection (WRED), use a command of the form shown in the following table. This example is based on "Device2" in the network topology shown in the figure in the configuration examples section.

SUMMARY STEPS

1. **show queueing interface** *subinterface*

DETAILED STEPS

show queueing interface *subinterface*

Example:

```
Device2# show queueing interface gigabitethernet6/0/0
```

Verifies the WRED configuration on the specified interface.

```
Device2# show queueing interface gigabitethernet6/0/0

Interface Gige6/0/0 queueing strategy:random early detection (WRED)
     Exp-weight-constant:9 (1/512)
     Mean queue depth:0

     Class   Random       Tail    Minimum    Maximum     Mark
               drop       drop  threshold  threshold  probability
       0         85          0         20         40     1/10
       1         22          0         22         40     1/10
       2          0          0         24         40     1/10
       3          0          0         26         40     1/10
       4          0          0         28         40     1/10
       5          0          0         31         40     1/10
       6          0          0         33         40     1/10
       7          0          0         35         40     1/10
      rsvp        0          0         37         40     1/10
```

Configuring CAR

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *name*
4. **rate-limit input** [**access-group** [**rate-limit**] *acl-index*] *bps burst-normal burst-max* **conform-action** *conform-action* **exceed-action** *exceed-action*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *name* **Example:** `Device(config)# interface gigabitethernet`	Designates the input interface, and enters interface configuration mode.
Step 4	**rate-limit input** [**access-group** [**rate-limit**] *acl-index*] *bps burst-normal burst-max* **conform-action** *conform-action* **exceed-action** *exceed-action* **Example:** `Device(config-if)# rate-limit input access-group 101 496000 32000 64000 conform-action set-prec-transmit 4`	Specifies the action to take on packets during label imposition.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Verifying the CAR Configuration

SUMMARY STEPS

1. **show interfaces** *slot/port* **rate-limit**

DETAILED STEPS

show interfaces *slot/port* **rate-limit**

Example:

```
Device2# show interfaces fe1/1/1 rate-limit
```

Verifies the CAR configuration, use a command of the following form.

```
Device2# show interfaces fe1/1/1 rate-limit

FastEthernet1/1/1
   Input
     matches:access-group 101
       params: 496000 bps, 32000 limit, 64000 extended limit
       conformed 2137 packets, 576990 bytes; action:set-prec-transmit 4
```

```
exceeded 363 packets, 98010 bytes; action:set-prec-transmit 0
last packet:11788ms ago, current burst:39056 bytes
last cleared 00:01:18 ago, conformed 58000 bps, exceeded 10000 bps
```

Configuring CBWFQ

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-map-name*
4. **match** *type number*
5. **policy-map** *policy-map-name*
6. **class** *class-map-name*
7. **bandwidth** *number*
8. **interface** *type number*
9. **service-policy output** *policy-map-name*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map** *class-map-name* **Example:** `Device(config)# class-map class-map-1`	Creates a class map, and enters class-map configuration mode.
Step 4	**match** *type number* **Example:** `Device(config-cmap)# match ip precedence 0 1`	Specifies the traffic on which the class map is to match.
Step 5	**policy-map** *policy-map-name* **Example:** `Device(config-cmap)# policy-map outputmap`	Creates a policy map, and enters policy-map configuration mode.
Step 6	**class** *class-map-name* **Example:**	Associates the class map with the policy map.

	Command or Action	Purpose
	Device(config-pmap)# class class-map-1	
Step 7	**bandwidth** *number* **Example:** `Device(config-pmap-c)# bandwidth 10000`	Associates the bandwidth (CBWFQ) action to act on traffic matched by the class map, and enters policy-map class configuration mode.
Step 8	**interface** *type number* **Example:** `Device(config-pmap-c)# interface gigabitethernet0/0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 9	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output outputmap`	Assigns the policy map to an interface.
Step 10	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Verifying the CBWFQ Configuration

SUMMARY STEPS

1. **show policy-map interface** *type number*

DETAILED STEPS

show policy-map interface *type number*

Example:

```
Device5# show policy-map interface fe5/1/0
```

Verifies the class-based weighted fair queueing (CBWFQ) configuration, use a command of the following form. This example is based on “Device 5” in the network topology shown in the figure in the configuration examples section.

```
Device5# show policy-map interface fe5/1/0

  FastEthernet5/1/0
   service-policy output:outputmap
     class-map:prec_01 (match-all)
       522 packets, 322836 bytes
       5 minute rate 1000 bps
       match:ip precedence 0  1
       queue size 0, queue limit 1356
       packet output 522, packet drop 0
       tail/random drop 0, no buffer drop 0, other drop 0
       bandwidth:class-based wfq, weight 10
       random-detect:
         Exp-weight-constant:9 (1/512)
         Mean queue depth:0
```

```
Class Random       Tail    Minimum    Maximum     Mark      Output
        drop       drop  threshold  threshold  probability packets
0         0          0       3390       6780      1/10         522
1         0          0       3813       6780      1/10           0
2         0          0       4236       6780      1/10           0
3         0          0       4659       6780      1/10           0
4         0          0       5082       6780      1/10           0
5         0          0       5505       6780      1/10           0
6         0          0       5928       6780      1/10           0
7         0          0       6351       6780      1/10           0

  class-map:prec_23 (match-all)
    0 packets, 0 bytes
    5 minute rate 0 bps
    match:ip precedence 2  3
    queue size 0, queue limit 0
    packet output 0, packet drop 0
    tail/random drop 0, no buffer drop 0, other drop 0
    bandwidth:class-based wfq, weight 15
    random-detect:
      Exp-weight-constant:9 (1/512)
      Mean queue depth:0
Class Random       Tail    Minimum    Maximum     Mark      Output
        drop       drop  threshold  threshold  probability packets
0         0          0          0          0    1/10             0
1         0          0          0          0    1/10             0
2         0          0          0          0    1/10             0
3         0          0          0          0    1/10             0
4         0          0          0          0    1/10             0
5         0          0          0          0    1/10             0
6         0          0          0          0    1/10             0
7         0          0          0          0    1/10             0

  class-map:prec_45 (match-all)
    2137 packets, 576990 bytes
    5 minute rate 16000 bps
    match:ip precedence 4  5
    queue size 0, queue limit 2712
    packet output 2137, packet drop 0
    tail/random drop 0, no buffer drop 0, other drop 0
    bandwidth:class-based wfq, weight 20
    random-detect:
      Exp-weight-constant:9 (1/512)
      Mean queue depth:0
Class Random       Tail    Minimum    Maximum     Mark      Output
        drop       drop  threshold  threshold  probability packets
0         0          0       3390       6780    1/10             0
1         0          0       3813       6780    1/10             0
2         0          0       4236       6780    1/10             0
3         0          0       4659       6780    1/10             0
4         0          0       5082       6780    1/10          2137
5         0          0       5505       6780    1/10             0
6         0          0       5928       6780    1/10             0
7         0          0       6351       6780    1/10             0

  class-map:prec_67 (match-all)
    0 packets, 0 bytes
    5 minute rate 0 bps
    match:ip precedence 6  7
    queue size 0, queue limit 0
    packet output 0, packet drop 0
    tail/random drop 0, no buffer drop 0, other drop 0
    bandwidth:class-based wfq, weight 25
    random-detect:
```

```
        Exp-weight-constant:9 (1/512)
        Mean queue depth:0
  Class Random       Tail    Minimum    Maximum     Mark      Output
          drop       drop  threshold  threshold  probability packets
  0          0          0          0          0    1/10            0
  1          0          0          0          0    1/10            0
  2          0          0          0          0    1/10            0
  3          0          0          0          0    1/10            0
  4          0          0          0          0    1/10            0
  5          0          0          0          0    1/10            0
  6          0          0          0          0    1/10            0
  7          0          0          0          0    1/10            0

    class-map:class-default (match-any)
      0 packets, 0 bytes
      5 minute rate 0 bps
      match:any
        0 packets, 0 bytes
        5 minute rate 0 bps
      queue size 0, queue limit 4068
      packet output 90, packet drop 0
      tail/random drop 0, no buffer drop 0, other drop 0
Device5#
Device5# show queueing interface fa1/1/0

Interface FastEthernet1/1/0 queueing strategy:VIP-based fair queueing
  FastEthernet1/1/0 queue size 0
        pkts output 2756, wfq drops 0, nobuffer drops 0
  WFQ:aggregate queue limit 13561 max available buffers 13561

      Class 0:weight 30 limit 4068 qsize 0 pkts output 97 drops 0
      Class 2:weight 10 limit 1356 qsize 0 pkts output 522 drops 0
      Class 3:weight 15 limit 0 qsize 0 pkts output 0 drops 0
      Class 4:weight 20 limit 2712 qsize 0 pkts output 2137 drops 0
      Class 5:weight 25 limit 0 qsize 0 pkts output 0 drops 0 \
```

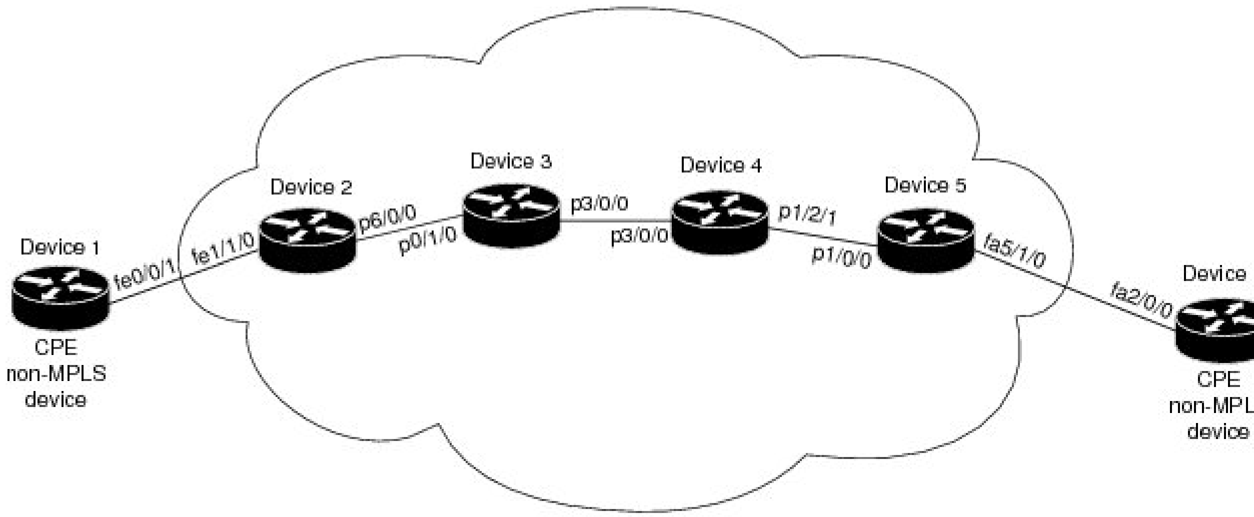
What to do next

-

Configuration Examples for MPLS Quality of Service

The configuration examples are based on the sample network topology shown in the figure below.

Figure 21: Sample Network Topology for Configuring MPLS CoS on Device Interfaces



Example: Configuring Cisco Express Forwarding

Cisco Express Forwarding must be running on all devices in the Multiprotocol Label Switching (MPLS) network for MPLS CoS to work. To enable Cisco Express Forwarding, use one of the following commands:

```
Device(config)# ip cef
```

or

```
Device(config)# ip cef distributed
```

Example: Running IP on Device 1

The following commands enable IP routing on Device 1. All devices in the figure must have IP enabled. Device 1 is not part of the Multiprotocol Label Switching (MPLS) network.

```
!
ip routing
!
hostname R1
!
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
interface FastEthernet0/0/1
 ip address 10.0.0.1 255.0.0.0
!
router ospf 100
 network 10.0.0.0 0.255.255.255 area 100
 network 10.0.0.1 0.255.255.255 area 100
```

Example: Running MPLS on Device 2

Device 2 is a label edge router. Cisco Express Forwarding and Multiprotocol Label Switching (MPLS) must be enabled on this device. Committed access rate (CAR) is also configured on Device 2 and Fast Ethernet interface 1/1/3. The CAR policy used at Fast Ethernet interface 1/1/0 acts on incoming traffic matching access-list 101. If the traffic rate is less than the committed information rate (in this example, 496000), the traffic will be sent with IP precedence 4. Otherwise, this traffic will be sent with IP precedence 0.

```
!
ip routing
!
hostname R2
!
ip cef
mpls ip
tag-switching advertise-tags
!
interface Loopback0
 ip address 10.10.10.10 255.255.255.255
!
interface FastEthernet1/1/0
 ip address 10.0.0.2 255.0.0.0
 rate-limit input access-group 101 496000 32000 64000 conform-action set-prec-transmit 4
exceed-action set-prec-transmit 0
!
interface POS6/0/0
 ip address 10.0.0.1 255.0.0.0
 mpls label protocol ldp
 mpls ip
 random-detect
 clock source internal
!
router ospf 100
 network 10.0.0.0 0.255.255.255 area 100
 network 10.1.0.0 0.255.255.255 area 100
 network 11.0.1.0 0.255.255.255 area 100
!
access-list 101 permit ip host 10.10.1.1 any
```

Example: Running MPLS on Device 3

Device 3 is running Multiprotocol Label Switching (MPLS). Cisco Express Forwarding and MPLS must be enabled on this device.

```
!
ip routing
mpls ip
tag-switching advertise-tags
!
hostname R3
!
interface Loopback0
 ip address 10.10.10.10 255.255.255.255
!
interface POS0/1/0
 ip address 10.0.0.2 255.0.0.0
 mpls label protocol ldp
 mpls ip
```

```
 crc 16
!
interface POS3/0/0
 ip address 10.0.0.1 255.0.0.0
 mpls label protocol ldp
 mpls ip
 crc 16
 clock source internal
 tx-cos stm16-rx
!
router ospf 100
 network 10.0.1.0 0.255.255.255 area 100
 network 10.0.0.1 0.255.255.255 area 100
 network 10.1.0.0 0.255.255.255 area 100
!
cos-queue-group stm16-rx
 precedence 0 random-detect-label 0
 precedence 0 queue 0
 precedence 1 queue 1
 precedence 1 random-detect-label 1
 precedence 2 queue 2
 precedence 2 random-detect-label 2
 precedence 3 random-detect-label 2
 precedence 4 random-detect-label 2
 precedence 5 random-detect-label 2
 precedence 6 random-detect-label 2
 precedence 7 queue low-latency
 precedence 7 random-detect-label 2
 random-detect-label 0 250 1000 1
 random-detect-label 1 500 1250 1
 random-detect-label 2 750 1500 1
 queue 0 50
 queue 1 100
 queue 2 150
 queue low-latency alternate-priority 500
```

Example: Running MPLS on Device 4

Device 4 is running Multiprotocol Label Switching (MPLS). Cisco Express Forwarding and MPLS must be enabled on this device.

```
!
ip routing
mpls ip
tag-switching advertise-tags
!
hostname R4
!
interface Loopback0
 ip address 10.0.0.0 255.255.255.255
!
interface POS1/2/1
 ip address 10.0.0.1 255.0.0.0
 mpls label protocol ldp
 mpls ip
 crc 16
 clock source internal
 tx-cos stm16-rx
!
router ospf 100
 network 10.0.0.0 0.255.255.255 area 100
```

```
 network 10.1.0.0 0.255.255.255 area 100
 network 10.0.1.0 0.255.255.255 area 100
!
cos-queue-group stm16-rx
 precedence 0 queue 0
 precedence 0 random-detect-label 0
 precedence 1 queue 1
 precedence 1 random-detect-label 1
 precedence 2 queue 2
 precedence 2 random-detect-label 2
 precedence 3 random-detect-label 2
 precedence 4 random-detect-label 2
 precedence 5 random-detect-label 2
 precedence 6 random-detect-label 2
 precedence 7 queue low-latency
 random-detect-label 0 250 1000 1
 random-detect-label 1 500 1250 1
 random-detect-label 2 750 1500 1
 queue 0 50
 queue 1 100
 queue 2 150
 queue low-latency alternate-priority 200
```

Example: Running MPLS on Device 5

Device 5 is running Multiprotocol Label Switching (MPLS). Cisco Express Forwarding and MPLS must be enabled on this device. Device 5 has class-based weighted fair queueing (CBWFQ) enabled on Fast Ethernet interface 5/1/0. In this example, class maps are created, matching packets with various IP precedence values. These class maps are then used in a policy map named “outputmap,” where CBWFQ is assigned to each class. Finally, the policy map is assigned to the outbound Fast Ethernet interface 5/1/0.

```
!
ip routing
mpls ip
tag-switching advertise-tags
!
hostname R5
!
!
class-map match-all prec_01
  match ip precedence 0  1
class-map match-all prec_23
  match ip precedence 2  3
class-map match-all prec_45
  match ip precedence 4  5
class-map match-all prec_67
  match ip precedence 6  7
!
!
policy-map outputmap
  class prec_01
    bandwidth 10000
    random-detect
  class prec_23
    bandwidth 15000
    random-detect
  class prec_45
    bandwidth 20000
    random-detect
```

```
  class prec_67
    bandwidth 25000
    random-detect
!
ip cef distributed
!
interface Loopback0
 ip address 10.0.0.0 255.255.255.255
 no ip directed-broadcast
!
interface POS1/1/0
 ip address 10.0.0.2 255.0.0.0
 ip route-cache distributed
 mpls label protocol ldp
 mpls ip
!
interface FastEthernet5/1/0
 ip address 10.0.0.1 255.0.0.0
 ip route-cache distributed
 full-duplex
 service-policy output outputmap
!
router ospf 100
 network 10.1.0.0 0.255.255.255 area 100
 network 10.0.1.0 0.255.255.255 area 100
 network 10.0.0.1 0.255.255.255 area 100
```

Example: Running IP on Device 6

Device 6 is running IP. Cisco Express Forwarding must be enabled on this device. Device 6 is not part of the Multiprotocol Label Switching (MPLS) network.

```
!
ip routing
!
hostname R6
!
ip cef distributed
!
interface Loopback0
 ip address 10.0.0.0 255.255.255.255
!
interface FastEthernet2/0/0
 ip address 10.0.0.2 255.0.0.0
 ip route-cache distributed
 full-duplex
!
router ospf 100
 network 10.0.0.0 0.255.255.255 area 100
 network 10.1.0.0 0.255.255.255 area 100
!
```

Additional References for MPLS Quality of Service

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS QoS commands	Cisco IOS Quality of Service Solutions Command Reference Cisco IOS Multiprotocol Label Switching Command Reference

MIBs

MIB	MIBs Link
• CISCO-WRED-MIB • CISCO-CAR-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for MPLS Quality of Service

Table 18: Feature Information for MPLS Quality of Service

Feature Name	Releases	Feature Information
MPLS Quality of Service	12.0(5)T 12.0(11)T 12.0(22)S 12.2(17b)SXA 12.2(8)T Cisco IOS XE Release 2.1	The MPLS Quality of Service feature (formerly named as the MPLS CoS feature) enables you to provide differentiated services across an MPLS network. To satisfy a wide range of networking requirements, you can specify the class of service applicable to each transmitted IP packet. Different classes of service can be established for IP packets by setting the IP precedence bit in the header of each packet No new or modified commands were introduced.

CHAPTER 8

QoS Policy Support on L2VPN ATM PVPs

This feature enables you to configure Quality of Service (QoS) service policies in ATM permanent virtual path (PVP) mode for Layer 2 Virtual Private Networks (L2VPNs).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for QoS Policy Support on L2VPN ATM PVPs

Before configuring QoS policies on L2VPN ATM PVPs, you should understand the concepts and configuration instructions in the following documents:

- Any Transport over MPLS
- Applying QoS Features Using the MQC

Restrictions for QoS Policy Support on L2VPN ATM PVPs

- Queueing-based policies are not supported in ATM PVP mode and virtual circuit (VC) mode at the same time under the same main interface. However, nonqueueing policies can be mixed. For example, you can configure a nonqueueing policy in PVP mode and configure queueing policies on in VC mode under the same main interface. Similarly, you can configure a queueing policy in PVP mode and configure nonqueueing policies in VC mode in the input or output direction.
- ATM PVP mode does not support sessions.
- When you enable a policy in PVP mode, do not configure ATM rates on the VCs that are part of the PVP. The VCs should be unspecified bit rate (UBR) VCs only.
- If VCs are part of a PVP that has a policy configured, you cannot configure ATM VC traffic shaping.
- You cannot configure a queueing policy on an ATM PVP with UBR.
- You cannot configure queueing-based policies with UBR traffic shaping.

Information About QoS Policy Support on L2VPN ATM PVPs

The MQC Structure

The MQC structure allows you to define a traffic class, create a traffic policy, and attach the traffic policy to an interface.

The MQC structure consists of the following three high-level steps.

SUMMARY STEPS

1. Define a traffic class by using the **class-map**command. A traffic class is used to classify traffic.
2. Create a traffic policy by using the **policy-map** command. (The terms traffic policy and policy map are often synonymous.) A traffic policy (policy map) contains a traffic class and one or more QoS features that will be applied to the traffic class. The QoS features in the traffic policy determine how to treat the classified traffic.
3. Attach the traffic policy (policy map) to the interface by using the **service-policy** command.

DETAILED STEPS

Step 1 Define a traffic class by using the **class-map**command. A traffic class is used to classify traffic.

Step 2 Create a traffic policy by using the **policy-map** command. (The terms traffic policy and policy map are often synonymous.) A traffic policy (policy map) contains a traffic class and one or more QoS features that will be applied to the traffic class. The QoS features in the traffic policy determine how to treat the classified traffic.

Step 3 Attach the traffic policy (policy map) to the interface by using the **service-policy** command.

Elements of a Traffic Class

A traffic class contains three major elements: a traffic class name, a series of match commands, and, if more than one match command is used in the traffic class, instructions on how to evaluate these match commands.

The match commands are used for classifying packets. Packets are checked to determine whether they meet the criteria specified in the match commands; if a packet meets the specified criteria, that packet is considered a member of the class. Packets that fail to meet the matching criteria are classified as members of the default traffic class.

Elements of a Traffic Policy

A traffic policy contains three elements: a traffic policy name, a traffic class (specified with the class command), and the command used to enable the QoS feature.

The traffic policy (policy map) applies the enabled QoS feature to the traffic class once you attach the policy map to the interface (by using the service-policy command).

Note A packet can match only one traffic class within a traffic policy. If a packet matches more than one traffic class in the traffic policy, the first traffic class defined in the policy will be used.

How to Configure QoS Policy Support on L2VPN ATM PVPs

Enabling a Service Policy in ATM PVP Mode

You can enable a service policy in ATM PVP mode. You can also enable a service policy on PVP on a multipoint subinterface.

Note The **show policy-map interface** command does not display service policy information for ATM interfaces.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **atm pvp** *vpi* **l2transport**
5. **service-policy** [**input** | **output**] *policy-map-name*
6. **xconnect** *peer-router-id vcid* encapsulation mpls
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Defines the interface and enters interface configuration mode.
Step 4	**atm pvp** *vpi* **l2transport** **Example:** `Router(config-if)# atm pvp 1 l2transport`	Specifies that the PVP is dedicated to transporting ATM cells and enters l2transport PVP configuration mode. • The **l2transport**keyword indicates that the PVP is for cell relay. This mode is for Layer 2 transport only; it is not for regular PVPs.
Step 5	**service-policy** [**input** | **output**] *policy-map-name* **Example:** `Router(config-if-atm-l2trans-pvp)# service policy input pol1`	Enables a service policy on the specified PVP.
Step 6	**xconnect** *peer-router-id vcid* encapsulation mpls **Example:** `Router(config-if-atm-l2trans-pvp)# xconnect 10.0.0.1 123 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC. • The syntax for this command is the same as for all other Layer 2 transports.

Step 7	**end**
	Example:
	`Router(config-if-atm-l2trans-pvp)#`
	`end`
		Exits l2transport PVP configuration mode and returns to privileged EXEC mode.

Enabling a Service Policy in ATM PVP Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

You can enable a service policy in ATM PVP mode. You can also enable a service policy on PVP on a multipoint subinterface.

Note The **show policy-map interface** command does not display service policy information for ATM interfaces.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **atm pvp** *vpi* **l2transport**
5. **service-policy** [**input** | **output**] *policy-map-name*
6. **end**
7. **interface pseudowire** *number*
8. **encapsulation mpls**
9. **neighbor** *peer-address vcid-value*
10. **exit**
11. **l2vpn xconnect context** *context-name*
12. **member pseudowire** *interface-number*
13. **member gigabitethernet** *interface-number*
14. **end**
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Defines the interface and enters interface configuration mode.
Step 4	**atm pvp** *vpi* **l2transport** **Example:** `Router(config-if)# atm pvp 1 l2transport`	Specifies that the PVP is dedicated to transporting ATM cells and enters l2transport PVP configuration mode. • The **l2transport**keyword indicates that the PVP is for cell relay. This mode is for Layer 2 transport only; it is not for regular PVPs.

	Command or Action	Purpose
Step 5	**service-policy** [**input** | **output**] *policy-map-name* **Example:** `Router(config-if-atm-l2trans-pvp)# service policy input pol1`	Enables a service policy on the specified PVP.
Step 6	**end** **Example:** `Router(config-if-atm-l2trans-pvp)# end`	Exits to privileged EXEC mode.
Step 7	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 8	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 9	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 13	**member gigabitethernet** *interface-number* **Example:** `Router(config-xconnect)# member GigabitEthernet0/0/0.1`	Specifies the location of the Gigabit Ethernet member interface.

	Command or Action	Purpose
Step 14	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Step 15	**end**
	Example:
	`Router(config-xconnect)#`
	`end`
		Exits xconnecrt configuration mode and returns to privileged EXEC mode.

Enabling Traffic Shaping in ATM PVP Mode

Traffic shaping commands are supported in PVP mode. For egress VP shaping, one configuration command is supported for each ATM service category. The supported service categories are constant bit rate (CBR), UBR, variable bit rate-nonreal time (VBR-NRT), and variable bit rate real-time(VBR-RT).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **atm pvp** *vpi* **l2transport**
5. Do one of the following:
   - **ubr** *pcr*
   -
   - **cbr** *pcr*
   - or
   - **vbr-nrt** *pcr* *scr mbs*
   - or
   - **vbr-rt** *pcr scr mbs*
6. **xconnect** *peer-router-id vcid* encapsulation mpls

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**interface atm** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Defines the interface and enters interface configuration mode.
Step 4	**atm pvp** *vpi* **l2transport** **Example:** `Router(config-if)# atm pvp 1 l2transport`	Specifies that the PVP is dedicated to transporting ATM cells and enters l2transport PVP configuration mode. • The **l2transport**keyword indicates that the PVP is for cell relay. This mode is for Layer 2 transport only; it is not for regular PVPs.
Step 5	Do one of the following: • **ubr** *pcr* • • **cbr** *pcr* • or • **vbr-nrt** *pcr* *scr mbs* • or • **vbr-rt** *pcr scr mbs* **Example:** `Router(config-if-atm-l2trans-pvp)# cbr 1000`	Enables traffic shaping in ATM PVP mode. • *pcr* = peak cell rate • *scr* = sustain cell rate • *mbs* = maximum burst size
Step 6	**xconnect** *peer-router-id vcid* encapsulation mpls **Example:** `Router(config-if-atm-l2trans-pvp)# xconnect 10.0.0.1 123 encapsulation mpls`	Binds the attachment circuit to a pseudowire VC. • The syntax for this command is the same as for all other Layer 2 transports.

Enabling Traffic Shaping in ATM PVP Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

Traffic shaping commands are supported in PVP mode. For egress VP shaping, one configuration command is supported for each ATM service category. The supported service categories are constant bit rate (CBR), UBR, variable bit rate-nonreal time (VBR-NRT), and variable bit rate real-time(VBR-RT).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot* / *subslot* / *port* [**.** *subinterface*]
4. **atm pvp** *vpi* **l2transport**

5. Do one of the following:
   - **ubr** *pcr*
   -
   - **cbr** *pcr*
   - or
   - **vbr-nrt** *pcr* *scr mbs*
   - or
   - **vbr-rt** *pcr scr mbs*
6. **end**
7. **interface pseudowire** *number*
8. **encapsulation mpls**
9. **neighbor** *peer-address vcid-value*
10. **exit**
11. **l2vpn xconnect context** *context-name*
12. **member pseudowire** *interface-number*
13. **member gigabitethernet** *interface-number*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot* / *subslot* / *port* [**.** *subinterface*] **Example:** `Router(config)# interface atm1/0/0`	Defines the interface and enters interface configuration mode.
Step 4	**atm pvp** *vpi* **l2transport** **Example:** `Router(config-if)# atm pvp 1 l2transport`	Specifies that the PVP is dedicated to transporting ATM cells and enters l2transport PVP configuration mode. • The **l2transport**keyword indicates that the PVP is for cell relay. This mode is for Layer 2 transport only; it is not for regular PVPs.
Step 5	Do one of the following: • **ubr** *pcr* •	Enables traffic shaping in ATM PVP mode. • *pcr* = peak cell rate • *scr* = sustain cell rate

	Command or Action	Purpose
	• **cbr** *pcr* • or • **vbr-nrt** *pcr* *scr mbs* • or • **vbr-rt** *pcr scr mbs* **Example:** `Router(config-if-atm-l2trans-pvp)# cbr 1000`	• *mbs* = maximum burst size
Step 6	**end** **Example:** `Router(config-if-atm-l2trans-pvp)# end`	Exits to privileged EXEC mode.
Step 7	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 8	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 9	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 13	**member gigabitethernet** *interface-number* **Example:**	Specifies the location of the Gigabit Ethernet member interface.

	Command or Action	Purpose
	`Router(config-xconnect)# member GigabitEthernet0/0/0.1`	
Step 14	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Enabling Traffic Shaping in ATM PVP Mode Example using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example enables traffic shaping in ATM PMP mode.

```
interface atm 1/0
 atm pvp 100 l2transport
  ubr 1000
  xconnect 10.11.11.11 777 encapsulation mpls
 atm pvp 101 l2transport
  cbr 1000
  xconnect 10.11.11.11 888 encapsulation mpls
 atm pvp 102 l2transport
  vbr-nrt 1200 800 128
  xconnect 10.11.11.11 999 encapsulation mpls
```

Enabling Matching of ATM VCIs

You can match on an ATM VCI or range of VCIs, using the **match atm-vci** command in class-map configuration mode.

Note When you configure the **match atm-vci**command in class-map configuration mode, you can add this class map to a policy map that can be attached only to an ATM VP.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-map-name* [**match-all** | **match-any**]
4. **match atm-vci** *vc-id* **[- vc-id]**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** *class-map-name* [**match-all** | **match-any**] **Example:** `Router(config)# class-map class1`	Creates a class map to be used for matching traffic to a specified class, and enters class-map configuration mode.
Step 4	**match atm-vci** *vc-id* **[- vc-id]** **Example:** `Router(config-cmap)# match atm-vci 50`	Enables packet matching on an ATM VCI or range of VCIs. The range is 32 to 65535. **Note** You can use the **match not** command to remove the match criteria.
Step 5	**end** **Example:** `Router(config-cmap)# end`	(Optional) Returns to privileged EXEC mode.

Configuration Examples for QoS Policy Support on L2VPN ATM PVPs

Example Enabling Traffic Shaping in ATM PVP Mode

The following example enables traffic shaping in ATM PMP mode.

```
int atm 1/0/0
   atm pvp 100 l2transport
       ubr 1000
       xconnect 10.11.11.11 777 encapsulation mpls
   atm pvp 101 l2transport
       cbr 1000
       xconnect 10.11.11.11 888 encapsulation mpls
   atm pvp 102 l2transport
       vbr-nrt 1200 800 128
       xconnect 10.11.11.11 999 encapsulation mpls
```

Example Enabling Traffic Shaping in ATM PVP Mode using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example enables traffic shaping in ATM PMP mode.

```
int atm 1/0/0
   atm pvp 100 l2transport
       ubr 1000
       interface pseudowire 100
       encapsulation mpls
       neighbor 10.0.0.1 123
!
l2vpn xconnect context A
 member pseudowire 100
 member g0/0/0.1
   atm pvp 101 l2transport
       cbr 1000
       interface pseudowire 100
       encapsulation mpls
       neighbor 10.0.0.1 123
!
l2vpn xconnect context A
 member pseudowire 100
 member g0/0/0.1
   atm pvp 102 l2transport
       vbr-nrt 1200 800 128
       interface pseudowire 100
       encapsulation mpls
       neighbor 10.0.0.1 123
!
l2vpn xconnect context A
 member pseudowire 100
 member g0/0/0.1
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Description of commands associated with MPLS and MPLS applications	*Cisco IOS Multiprotocol Label Switching Command Reference*
Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC)	Applying QoS Features Using the MQC
Any Transport over MPLS	Any Transport over MPLS

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for QoS Policy Support on L2VPN ATM PVPs

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 19: Feature Information for QoS Policy Support on L2VPN ATM PVPs

Feature Name	Releases	Feature Information
QoS Policy Support on L2VPN ATM PVPs	Cisco IOS XE Release 2.3	This feature enables you to configure Quality of Service (QoS) service policies in ATM permanent virtual path (PVP) mode for Layer 2 Virtual Private Networks (L2VPNs). The following commands were introduced or modified: **cbr**, **match atm-vci**, **service-policy**, **ubr**, **vbr-nrt**, **vbr-rt**.
Cell-Based ATM Shaping per PVP	Cisco IOS XE Release 2.3	This feature was introduced for Cisco ASR 1000 Series Aggregation Services Routers.

CHAPTER 9

MPLS Pseudowire Status Signaling

The MPLS Pseudowire Status Signaling feature enables you to configure the router so it can send pseudowire status to a peer router, even when the attachment circuit is down.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for MPLS Pseudowire Status Signaling

- Before configuring this feature, make sure that both peer routers are capable of sending and receiving pseudowire status messages.

Restrictions for MPLS Pseudowire Status Signaling

- Both peer routers must support the ability to send and receive pseudowire status messages in label advertisement and label notification messages. If both peer routers do not support pseudowire status messages, Cisco recommends that you disable the messages with the **no status** command.

- This feature is not integrated with Any Transport over MPLS (AToM) Virtual Circuit Connection Verification (VCCV).
- This feature is not integrated with Bidirectional Forwarding Detection (BFD).
- The standby and required switchover values from IETF draft-muley-pwe3-redundancy-02.txt are not supported.

Information About MPLS Pseudowire Status Signaling

How MPLS Pseudowire Status Switching Works

The pseudowire status messages are sent in label advertisement and label notification messages if the peer also supports the MPLS Pseudowire Status Signaling feature. You can issue the **show mpls l2transport vc detail** command to show that both the local and remote routers support pseudowire status messages. The following example shows the line of output to look for:

```
Router# show mpls l2transport vc detail
.
.
.
status TLV support (local/remote): enabled/supported
```

How MPLS Pseudowire Status Switching Works using the commands associated with the L2VPN Protocol-Based CLIs feature

The pseudowire status messages are sent in label advertisement and label notification messages if the peer also supports the MPLS Pseudowire Status Signaling feature. You can issue the **show l2vpn atom vc detail** command to show that both the local and remote routers support pseudowire status messages. The following example shows the line of output to look for:

```
Device# show l2vpn atom vc detail
.
.
.
status TLV support (local/remote): enabled/supported
```

When One Router Does Not Support MPLS Pseudowire Status Signaling

The peer routers must support the ability to send and receive pseudowire status messages in label advertisement and label notification messages. If one router does not support pseudowire status messages, Cisco recommends that you disable the messages with the **no status** command. This returns the router to label withdraw mode.

If the peer does not support the MPLS Pseudowire Status Signaling feature, the local router changes its mode of operation to label withdraw mode. You can issue the **show mpls l2transport vc detail** command to show that the remote router does not support pseudowire status messages. The following example shows the line of output to look for:

```
Router# show mpls l2transport vc detail

.

.

.

status TLV support (local/remote): enabled/not supported
```

When you issue the following **debug mpls l2transport vc**commands, the messages show that the peer router does not supportthe MPLS Pseudowire Status Signaling feature and that the local router is changing to withdraw mode, as shown in bold in the following example:

Router# debug mpls l2transport vc event Router# **debug mpls l2transport vc status event** Router# **debug mpls l2transport vc status fsm** Router# **debug mpls l2transport vc ldp**

*Feb 26 13:41:40.707: AToM LDP [10.1.1.2]: Sending label withdraw msg *Feb 26 13:41:40.707: AToM LDP [10.1.1.2]: VC Type 5, mtu 1500 *Feb 26 13:41:40.707: AToM LDP [10.1.1.2]: VC ID 100, label 18 *Feb 26 13:41:40.707: AToM LDP [10.1.1.2]: Status 0x0000000A [PW Status NOT supported]

When One Router Does Not Support MPLS Pseudowire Status Signaling using the commands associated with the L2VPN Protocol-Based CLIs feature

The peer routers must support the ability to send and receive pseudowire status messages in label advertisement and label notification messages. If one router does not support pseudowire status messages, we recommend that you disable the messages with the **no status** command. This returns the router to label withdraw mode.

If the peer does not support the MPLS Pseudowire Status Signaling feature, the local router changes its mode of operation to label withdraw mode. You can issue the **show l2vpn atom vc detail** command to show that the remote router does not support pseudowire status messages. The following example shows the line of output to look for:

```
Device# show l2vpn atom vc detail

.

.
```

```
.

status TLV support (local/remote): enabled/not supported
```

When you issue the following **debug l2vpn atom vc** commands, the messages show that the peer router does not support the MPLS Pseudowire Status Signaling feature and that the local router is changing to withdraw mode, as shown in the following example:

```
Device# debug l2vpn atom vc event
Device# debug l2vpn atom vc status event
Device# debug l2vpn atom vc status fsm
Device# debug l2vpn atom vc ldp

*Feb 26 13:41:40.707: AToM LDP [110.1.1.2]: Sending label withdraw msg
*Feb 26 13:41:40.707: AToM LDP [110.1.1.2]:   VC Type   5, mtu 1500
*Feb 26 13:41:40.707: AToM LDP [110.1.1.2]:   VC ID     100, label 18
*Feb 26 13:41:40.707: AToM LDP [110.1.1.2]:   Status 0x0000000A [PW Status NOT supported]
```

Status Messages Indicating That the Attachment Circuit Is Down

When the attachment circuit is down between the two routers, the output of the **show mpls l2transport vc detail** command shows the following status:

```
Router# show mpls l2transport vc detail

.

.

.

Last remote LDP TLV    status rcvd: AC DOWN(rx,tx faults)
```

The debug messages also indicate that the attachment circuit is down, as shown in bold in the command output:

Router# debug mpls l2transport vc event Router# **debug mpls l2transport vc status event** Router# **debug mpls l2transport vc status fsm** Router# **debug mpls l2transport vc ldp**

```
*Feb 26 11:51:42.427: AToM LDP [10.1.1.1]: Received notif msg, id 88
*Feb 26 11:51:42.427: AToM LDP [10.1.1.1]:   Status    0x00000007 [PW Status]
*Feb 26 11:51:42.427: AToM LDP [10.1.1.1]:   PW Status 0x00000006 [AC DOWN(rx,tx faults)]
```

Other pseudowire status messages include not-forwarding, pw-tx-fault, and pw-rx-fault.

Status Messages Indicating That the Attachment Circuit Is Down using the commands associated with the L2VPN Protocol-Based CLIs feature

When the attachment circuit is down between the two routers, the output of the **show l2vpn atom vc detail** command shows the following status:

```
Device# show l2vpn atom vc detail

.

.

.

Last remote LDP TLV    status rcvd: AC DOWN(rx,tx faults)
```

The debug messages also indicate that the attachment circuit is down, as shown in bold in the command output:

```
Device# debug l2vpn atom  vc event
Device# debug l2vpn atom  vc status event
Device# debug l2vpn atom vc status fsm
Device# debug l2vpn atom vc ldp



*Feb 26 11:51:42.427: AToM LDP [10.1.1.1]: Received notif msg, id 88
*Feb 26 11:51:42.427: AToM LDP [10.1.1.1]:   Status    0x00000007 [PW Status]
*Feb 26 11:51:42.427: AToM LDP [10.1.1.1]:   PW Status 0x00000006 [AC DOWN(rx,tx faults)]
```

Other pseudowire status messages include not-forwarding, pw-tx-fault, and pw-rx-fault.

Message Codes in the Pseudowire Status Messages

The **debug mpls l2transport vc**and the **show mpls l2transport vc detail** commands show output that contains message codes. For example:

```
    Label/status state machine: established, LruRru

AToM MGR [10.9.9.9, 100]: S:Evt local up, LndRru->LnuRru
```

The message codes (LruRru, LndRru, and LnuRru) indicate the status of the local and remote routers. You can use the following key to interpret the message codes:

- L--local router
- R--remote router
- r or n--ready (r) or not ready (n)
- u or d--up (u) or down (d) status

The output also includes other values:

- D--Dataplane
- S--Local shutdown

Message Codes in the Pseudowire Status Messages using the commands associated with the L2VPN Protocol-Based CLIs feature

The **debug l2vpn atom vc** and the **show l2vpn atom vc detail** commands show output that contains message codes. For example:

```
    Label/status state machine: established, LruRru

AToM MGR [10.9.9.9, 100]: S:Evt local up, LndRru->LnuRru
```

The message codes (LruRru, LndRru, and LnuRru) indicate the status of the local and remote routers. You can use the following key to interpret the message codes:

L—local router

R—remote router

r or n—ready (r) or not ready (n)

u or d—up (u) or down (d) status

The output also includes other values:

D—Dataplane

S—Local shutdown

How to Configure MPLS Pseudowire Status Signaling

Enabling MPLS Pseudowire Status Signaling

Perform the following task to enable the router to send pseudowire status to a peer router even when the attachment circuit is down. If both routers do not support pseudowire status messages, then disable the messages with the **no status** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *name*
4. **status**
5. **encapsulation mpls**
6. **exit**
7. **exit**
8. **show mpls l2transport vc detail**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** *name* **Example:** `Router(config)# pseudowire-class atom`	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 4	**status** **Example:** `Router(config-pw)# status`	(Optional) Enables the router to send pseudowire status messages to the peer router through label advertisement and label notification messages. **Note** By default, status messages are enabled. This step is included only in case status messages have been disabled. If you need to disable status messages because both peer routers do not support this functionality, enter the **no status**command.
Step 5	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 6	**exit** **Example:** `Router(config-pw)# exit`	Exits pseudowire class configuration mode.
Step 7	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 8	**show mpls l2transport vc detail** **Example:** `Router# show mpls l2transport vc detail`	Validates that pseudowire messages can be sent and received.

Enabling MPLS Pseudowire Status Signaling using the commands associated with the L2VPN Protocol-Based CLIs feature

Perform this task to enable the router to send pseudowire status to a peer router even when the attachment circuit is down. If both routers do not support pseudowire status messages, then disable the messages with the **no status** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *number*
4. **status**
5. **encapsulation mpls**
6. **neighbor** *peer-address vcid-value*
7. **exit**
8. **exit**
9. **show l2vpn atom vc detail**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 1`	Establishes an interface pseudowire with a value that you specify and enters pseudowire configuration mode.
Step 4	**status** **Example:** `Device(config-pw)# status`	(Optional) Enables the router to send pseudowire status messages to the peer router through label advertisement and label notification messages. **Note** By default, status messages are enabled. This step is included only in case status messages have been disabled. If you need to disable status messages because both peer routers do not support this functionality, enter the **no status**command.

	Command or Action	Purpose
Step 5	**encapsulation mpls** **Example:** `Device(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation.
Step 6	**neighbor** *peer-address vcid-value* **Example:** `Device(config-pw)#  neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 7	**exit** **Example:** `Device(config-pw)# exit`	Exits pseudowire class configuration mode.
Step 8	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.
Step 9	**show l2vpn atom vc detail** **Example:** `Device# show l2vpn atom vc detail`	Validates that pseudowire messages can be sent and received.

Configuration Examples for MPLS Pseudowire Status Signaling

Example MPLS Pseudowire Status Signaling

The following example configures the MPLS Pseudowire Status Signaling feature on two PE routers. By default, status messages are enabled. The **status** command is included in this example in case status messages have been disabled.

PE1

```
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
pseudowire-class atomstatus
 encapsulation mpls
 status
!
interface GigabitEthernet0/0/1
 xconnect 10.1.1.2 123 pw-class atomstatus
```

PE2

```
interface Loopback0
 ip address 10.1.1.2 255.255.255.255
!
pseudowire-class atomstatus
 encapsulation mpls
 status
!
interface GigabitEthernet3/3/0
 xconnect 10.1.1.1 123 pw-class atomstatus
```

Example MPLS Pseudowire Status Signaling using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example configures the MPLS Pseudowire Status Signaling feature on two PE routers. By default, status messages are enabled. The **status** command is included in this example in case status messages have been disabled.

PE1

```
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
template type pseudowire atomstatus
 encapsulation mpls
 status
!
interface pseudowire 100
 source template type pseudowire atomstatus
interface GigabitEthernet0/0/1
 service instance 300 ethernet
l2vpn xconnect context con1
 member GigabitEthernet2/1/1 service-instance 300
 member Pseudowire 100
```

PE2

```
interface Loopback0
 ip address 10.1.1.2 255.255.255.255
!
template type pseudowire atomstatus
 encapsulation mpls
 status
!
interface Pseudowire 100
 source template type pseudowire atomstatus
interface GigabitEthernet3/3/0
 service instance 300 ethernet
l2vpn xconnect context con1
 member GigabitEthernet2/1/1 service-instance 300
 member Pseudowire 100
```

Example Verifying That Both Routers Support Pseudowire Status Messages

You can issue the **show mpls l2transport vc detail** command to show that both the local and remote routers support pseudowire status messages. The following example shows the line of output to look for:

```
Router# show mpls l2transport vc detail
.
.
.
status TLV support (local/remote): enabled/supported
```

Example Verifying That Both Routers Support Pseudowire Status Messages using the commands associated with the L2VPN Protocol-Based CLIs feature

You can issue the **show l2vpn atom vc detail** command to show that both the local and remote routers support pseudowire status messages. The following example shows the line of output to look for:

```
Device# show l2vpn atom vc detail
.
.
.
status TLV support (local/remote): enabled/supported
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Description of commands associated with MPLS and MPLS applications	*Cisco IOS Multiprotocol Label Switching Command Reference*
Any Transport over MPLS	Any Transport over MPLS

Standards

Standard	Title
draft-ietf-pwe3-control-protocol-15.txt	Pseudowire Setup and Maintenance Using LDP
draft-ietf-pwe3-iana-allocation-08.txt	IANA Allocations for Pseudo Wire Edge to Edge Emulation (PWE3)
draft-martini-pwe3-pw-switching-03.txt	Pseudo Wire Switching

MIBs

MIB	MIBs Link
Pseudowire Emulation Edge-to-Edge MIBs for Ethernet, Frame Relay, and ATM Services	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for MPLS Pseudowire Status Signaling

Table 20: Feature Information for MPLS Pseudowire Status Signaling

Feature Name	Releases	Feature Information
MPLS Pseudowire Status Signaling	Cisco IOS XE Release 2.3	The MPLS Pseudowire Status Signaling feature enables you to configure the router so it can send pseudowire status to a peer router, even when the attachment circuit is down. The following commands were introduced or modified: **debug mpls l2transport vc**, **show mpls l2transport vc**, **status** (pseudowire class).

CHAPTER 10

L2VPN VPLS Inter-AS Option B

The L2VPN VPLS Inter-AS Option B feature expands the existing features of VPLS autodiscovery to operate across multiple Border Gateway Protocol (BGP) autonomous systems. Using BGP-based autodiscovery as the underlying framework, the L2VPN VPLS Inter-AS Option B feature creates a dynamic multisegmented pseudowire (PW) configuration between neighboring Autonomous System Boundary Routers (ASBRs.)

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for L2VPN VPLS Inter-AS Option B

The L2VPN VPLS Inter-AS Option B feature extends the functionality of the VPLS Autodiscovery: BGP Based feature. For example, as a result of L2VPN VPLS Inter-AS Option B feature, stateful switchover (SSO) and nonstop forwarding (NSF) are supported in a standard VPLS Autodiscovery configuration.

Before you configure the L2VPN VPLS Inter-AS Option B feature, enable the VPLS Autodiscovery: BGP Based feature and complete the steps described in the Modifying the VPLS Autodiscovery Settings for Use with L2VPN VPLS Inter-AS Option B, on page 356.

For more information about the VPLS Autodiscovery: BGP Based feature, see the "VPLS Autodiscovery: BGP" module.

Restrictions for L2VPN VPLS Inter-AS Option B

Introduced in Cisco IOS Release 15.1(1)S, the L2VPN VPLS Inter-AS Option B feature is supported only on a Cisco 7600 series router that is equipped with a line card capable of running Virtual Private LAN Switching (VPLS).

Information About L2VPN VPLS Inter-AS Option B

VPLS Functionality and L2VPN VPLS Inter-AS Option B

VPLS is a multipoint Layer 2 VPN (L2VPN) that connects two or more customer devices using Ethernet over Multiprotocol Label Switching (EoMPLS) bridging techniques.

VPLS Inter-AS support exists in a number of variations or options (for example, Option A, B, C, and D). The L2VPN VPLS Inter-AS Option B feature supports Option B only and is in compliance with RFC 4364, BGP/MPLS IP Virtual Private Networks (VPNs) .

For more information about VPLS, see the "VPLS Overview" section in the Configuring Multiprotocol Label Switching on the Optical Services Modules document.

L2VPN VPLS Inter-AS Option B Description

The L2VPN VPLS Inter-AS Option B feature extends VPLS across multiple autonomous system boundaries by dynamically creating multisegment pseudowires across the ASBRs.

When a router with external BGP (eBGP) advertises routes to its BGP neighbors, the router uses the source IP address as the next hop of the advertised routes.

When a router with internal BGP (iBGP) advertises routes to its BGP neighbors, the router does not change the next hop designation of the route advertised. For the L2VPN VPLS Inter-AS Option B feature, enter the **neighbor next-hop-self** command at the ASBRs. This forces the pseudowires to be targeted to the ASBR and not targeted to the provider edge (PE) routers. The net result is that a pseudowire for the first autonomous system is stitched to a pseudowire for the second autonomous system by means of a third pseudowire between the ASBRs. This creates a multisegmented pseudowire. For more information about multisegmented pseudowires, see the "L2VPN Multisegment Pseudowires" module.

Note The L2VPN VPLS Inter-AS Option B feature supports Route Processors (RPs), SSO, and NSF.

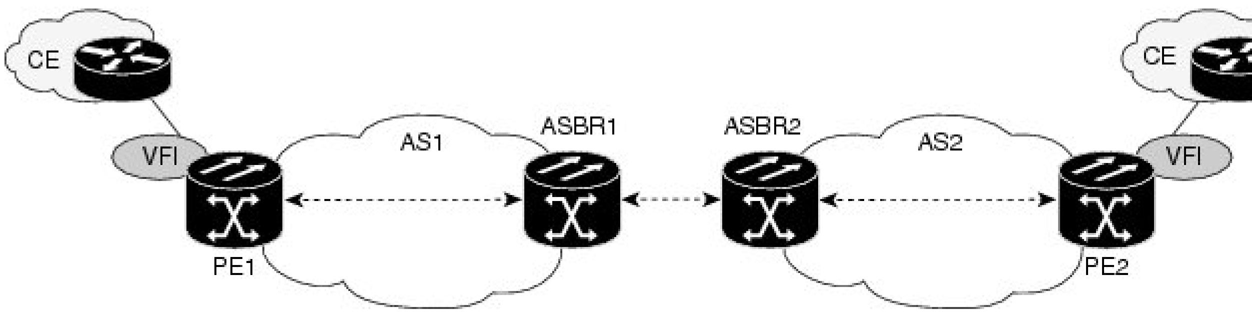
L2VPN VPLS Inter-AS Option B Sample Topology

The figure below illustrates a simplified L2VPN VPLS Inter-AS Option B topology. In this topology, AS1 and AS2 are the autonomous systems. ASBR1 and ASBR2 are ASBRs. A customer edge (CE) router is attached to both AS1 and AS2.

Each autonomous system consists of an ASBR and a PE router. PE1 belongs to a virtual forwarding instance (VFI) in AS1. PE2 belongs to a VFI in AS2. PE1 and PE2 are terminating PEs (TPEs).

Multisegmented pseudowires are created to establish dual connections between the TPE in the local ASBR to the TPE in the neighboring ASBR. The first segment establishes a path between the TPE in AS1 to ASBR1. The next segment establishes a path between the ASBR1 and ASBR2, and the final segment establishes a path between ASBR2 to the TPE in AS2.

Figure 22: Sample L2VPN VPLS Inter-AS Option B Topology



Active and Passive PEs in an L2VPN VPLS Inter-AS Option B Configuration

A TPE terminates a multisegment pseudowire. By default, the TPEs on both ends of a multisegmented pseudowire are in active mode. The L2VPN VPLS Inter-AS Option B feature requires that one of the TPEs be in passive mode. The system determines which PE is the passive TPE based on a comparison of the Target Attachment Individual Identifier (TAII) received from BGP and the Source Attachment Individual Identifier (SAII) of the local router. The TPE with the numerically higher identifier assumes the active role.

When you are configuring the PEs for the L2VPN VPLS Inter-AS Option B feature, use the **terminating-pe tie-breaker**command to negotiate the mode of the TPE. Then use the **mpls ldp discovery targeted-hello accept**command to ensure that a passive TPE can accept Label Distribution Protocol (LDP) sessions from the LDP peers.

For more information about configuring the PEs, see the Enabling L2VPN VPLS Inter-AS Option B on the Provider Edge (PE) Router, on page 365.

Benefits of L2VPN VPLS Inter-AS Option B

Private IP Addresses

While a large number of pseudowires are required, IPv4 reachability is maintained within the ASBR and, therefore, IP addresses are private.

One Targeted LDP Session

With the L2VPN VPLS Inter-AS Option B feature, only one targeted Label Distribution Protocol (LDP) session is created between the autonomous systems. Since only one targeted LDP session between autonomous systems is created, service providers can apply tighter security policies for control plane traffic going across the autonomous system.

How to Configure L2VPN VPLS Inter-AS Option B

Modifying the VPLS Autodiscovery Settings for Use with L2VPN VPLS Inter-AS Option B

Note Before you configure the L2VPN VPLS Inter-AS Option B feature, you must enable the VPLS Autodiscovery: BGP Based feature. Make sure you have enabled the VPLS Autodiscovery: BGP Based feature before proceeding with this task.

For the L2VPN VPLS Inter-AS Option B feature to function properly, you must configure the VPLS ID value and the route-target value for each PE router in the virtual forwarding instance (VFI). To modify these values, complete the following steps at each PE router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *vfi-name* **autodiscovery**
4. **vpn id** *vpn-id*
5. **vpls-id** {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*}
6. **route-target** [**import** | **export** | **both**] {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*}
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *vfi-name* **autodiscovery** **Example:** `Router(config)# l2 vfi vpls1 autodiscovery`	Enables the VPLS Autodiscovery: BGP Based feature on the PE router and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:**	Configures a VPN ID for the VPLS domain. • Enter a VPN ID value.

	Command or Action	Purpose
	`Router(config-vfi)# vpn id 10`	
Step 5	**vpls-id** {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*} **Example:** `Router(config-vfi)# vpls-id 5:300`	Specifies the VPLS ID. • The VPLS Autodiscovery: BGP Based feature automatically generates a VPLS ID using the BGP autonomous system number and the configured VFI VPN ID. Use this command to change the automatically generated VPLS ID for the PE in the VFI. • There are two formats for configuring the VPLS ID argument. It can be configured in the *autonomous-system-number* **:** *network number* (*ASN* **:** *nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address* **:** *nn)*.
Step 6	**route-target** [**import** | **export** | **both**] {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*} **Example:** `Router(config-vfi)# route-target 600:2222`	Specifies the route target (RT). • The VPLS Autodiscovery feature automatically generates a route target using the lower 6 bytes of the RD and VPN ID. Use this command to change the automatically generated route target for the PE in the VFI. • There are two formats for configuring the route target argument. It can be configured in the *autonomous-system-number* **:** *network number* (*ASN* **:** *nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address* **:** *nn)*.
Step 7	**exit** **Example:** `Router(config-vfi)# exit`	Exits L2 VFI configuration mode. • Commands take effect after the router exits L2 VFI configuration mode.

What to Do Next

Repeat the steps in the Modifying the VPLS Autodiscovery Settings for Use with L2VPN VPLS Inter-AS Option B, on page 356 at each PE in the autonomous system. Then proceed to the Enabling L2VPN VPLS Inter-AS Option B on the ASBR, on page 360.

Modifying the VPLS Autodiscovery Settings for Use with L2VPN VPLS Inter-AS Option B using the commands associated with the L2VPN Protocol-Based CLIs feature

Note Before you configure the L2VPN VPLS Inter-AS Option B feature, you must enable the VPLS Autodiscovery: BGP Based feature. Make sure you have enabled the VPLS Autodiscovery: BGP Based feature before proceeding with this task.

For the L2VPN VPLS Inter-AS Option B feature to function properly, you must configure the VPLS ID value and the route-target value for each PE router in the virtual forwarding instance (VFI). To modify these values, complete the following steps at each PE router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling ldp**
6. **vpls-id** {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*}
7. **route-target** [**import** | **export** | **both**] {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*}
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls1`	Establishes an L2VPN VFI context and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain. • Enter a VPN ID value.

	Command or Action	Purpose
Step 5	**autodiscovery bgp signaling ldp** **Example:** `Device(config-vfi)# autodiscovery bgp signaling ldp`	Enables the VPLS Autodiscovery: BGP Based feature on the PE router.
Step 6	**vpls-id** {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*} **Example:** `Device(config-vfi)# vpls-id 5:300`	Specifies the VPLS ID. • The VPLS Autodiscovery: BGP Based feature automatically generates a VPLS ID using the BGP autonomous system number and the configured VFI VPN ID. Use this command to change the automatically generated VPLS ID for the PE in the VFI. • There are two formats for configuring the VPLS ID argument. It can be configured in the *autonomous-system-number* **:** *network number* (*ASN* **:** *nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address* **:** *nn)*.
Step 7	**route-target** [**import** | **export** | **both**] {*autonomous-system-number* **:** *nn* | *ip-address* **:** *nn*} **Example:** `Device(config-vfi)# route-target 600:2222`	Specifies the route target (RT). • The VPLS Autodiscovery feature automatically generates a route target using the lower 6 bytes of the RD and VPN ID. Use this command to change the automatically generated route target for the PE in the VFI. • There are two formats for configuring the route target argument. It can be configured in the *autonomous-system-number* **:** *network number* (*ASN* **:** *nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address* **:** *nn)*.
Step 8	**exit** **Example:** `Device(config-vfi)# exit`	Exits L2 VFI configuration mode. • Commands take effect after the router exits L2 VFI configuration mode.

What to Do Next

Repeat the steps in the Modifying the VPLS Autodiscovery Settings for Use with L2VPN VPLS Inter-AS Option B, on page 356 at each PE in the autonomous system. Then proceed to the Enabling L2VPN VPLS Inter-AS Option B on the ASBR, on page 360.

Enabling L2VPN VPLS Inter-AS Option B on the ASBR

To enable the L2VPN VPLS Inter-AS Option B feature on the ASBR, complete the following steps on *each* ASBR in the autonomous system.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *peer-group-name*} **next-hop-self**
5. **address-family l2vpn vpls**
6. **no bgp default route-target filter**
7. **exit**
8. **exit**
9. **mpls ldp discovery targeted-hello accept**
10. Complete Steps 11 through 13, only if you are changing the range of VC IDs reserved for switching pseudowires. Otherwise, advance to Step 14.
11. **l2 pseudowire routing**
12. **switching-point vcid** *minimum-vcid-value maximum-vcid-value*
13. **exit**
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 1`	Configures the BGP routing process and enters router configuration mode. • Enter the number of the autonomous system.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **next-hop-self** **Example:** `Router(config-router)# neighbor 10.10.0.1 next-hop-self`	Configures the ASBR as the next hop for a BGP-speaking neighbor or peer group. • Enter the IP address or the peer group name. **Note** Use this command to identify each PE in the autonomous system.

	Command or Action	Purpose
Step 5	**address-family l2vpn vpls** **Example:** `Router(config-router)# address-family l2vpn vpls`	Configures a routing session using L2VPN endpoint provisioning address information and enters address family configuration mode.
Step 6	**no bgp default route-target filter** **Example:** `Router(config-router-af)# no bgp default route-target filter`	Enables pseudowire switching at the ASBR.
Step 7	**exit** **Example:** `Router(config-router-af) exit`	Exits address family configuration mode.
Step 8	**exit** **Example:** `Router(config-router) exit`	Exits router configuration mode.
Step 9	**mpls ldp discovery targeted-hello accept** **Example:** `Router(config)# mpls ldp discovery targeted-hello accept`	Configures the routers from which LDP sessions will be accepted. • With the **targeted-hello accept** keywords, LDP sessions from *any* router will be accepted. • For the other keyword choices available for this command, see the *Cisco IOS Multiprotocol Label Switching Command Reference*.
Step 10	Complete Steps 11 through 13, only if you are changing the range of VC IDs reserved for switching pseudowires. Otherwise, advance to Step 14.	
Step 11	**l2 pseudowire routing** **Example:** `Router(config))# l2 pseudowire routing`	(Optional) Enters Layer 2 pseudowire routing configuration mode.
Step 12	**switching-point vcid** *minimum-vcid-value maximum-vcid-value* **Example:** `Router(config-l2_pw_rtg)# switching-point vcid 200 3500`	(Optional) Configures a switching point and specifies a virtual circuit (VC) ID range. **Note** With the L2VPN VPLS Inter-AS Option B feature, VC IDs in the VC ID range of 1001 to 2147483647 are reserved for switching pseudowires. This command allows you to change this range if, for example, an existing xconnect VC is using one of the reserved VC IDs.

	Command or Action	Purpose
Step 13	**exit** **Example:** `Router(config-l2_pw_rtg)#) exit`	Exits Layer 2 pseudowire routing configuration mode.
Step 14	**end** **Example:** `Router(config)# end`	Exits global configuration mode.

What to Do Next

Repeat the steps in the Enabling L2VPN VPLS Inter-AS Option B on the ASBR, on page 360 at each ASBR in the autonomous system. Then proceed to the Enabling L2VPN VPLS Inter-AS Option B on the Provider Edge (PE) Router, on page 365.

Enabling L2VPN VPLS Inter-AS Option B on the ASBR using the commands associated with the L2VPN Protocol-Based CLIs feature

To enable the layer 2 virtual private network virtual private LAN services (L2VPN VPLS) Inter-AS Option B feature on the autonomous system boundary router (ASBR), perform this task on each ASBR in the autonomous system.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *peer-group-name*} **next-hop-self**
5. **address-family l2vpn vpls**
6. **no bgp default route-target filter**
7. **exit**
8. **exit**
9. **mpls ldp discovery targeted-hello accept**
10. Complete Steps 11 through 13, only if you are changing the range of VC IDs reserved for switching pseudowires. Otherwise, advance to Step 14.
11. **l2vpn**
12. **pseudowire routing**
13. **switching-point vcid** *minimum-vcid-value maximum-vcid-value*
14. **exit**
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 1`	Configures the BGP routing process and enters router configuration mode. • Enter the number of the autonomous system.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **next-hop-self** **Example:** `Device(config-router)# neighbor 10.10.0.1 next-hop-self`	Configures the ASBR as the next hop for a BGP-speaking neighbor or peer group. • Enter the IP address or the peer group name. **Note** Use this command to identify each PE in the autonomous system.
Step 5	**address-family l2vpn vpls** **Example:** `Device(config-router)# address-family l2vpn vpls`	Configures a routing session using L2VPN endpoint provisioning address information and enters address family configuration mode.
Step 6	**no bgp default route-target filter** **Example:** `Device(config-router-af)# no bgp default route-target filter`	Enables pseudowire switching at the ASBR.
Step 7	**exit** **Example:** `Device(config-router-af) exit`	Exits address family configuration mode.
Step 8	**exit** **Example:** `Device(config-router) exit`	Exits router configuration mode.
Step 9	**mpls ldp discovery targeted-hello accept** **Example:**	Configures the routers from which LDP sessions will be accepted.

	Command or Action	Purpose
	`Device(config)# mpls ldp discovery targeted-hello accept`	• With the **targeted-hello accept** keywords, LDP sessions from *any* router will be accepted. • For the other keyword choices available for this command, see the *Cisco IOS Multiprotocol Label Switching Command Reference*.
Step 10	Complete Steps 11 through 13, only if you are changing the range of VC IDs reserved for switching pseudowires. Otherwise, advance to Step 14.	
Step 11	**l2vpn** **Example:** `Device(config)# l2vpn`	(Optional) Enters Layer 2 VPN configuration mode.
Step 12	**pseudowire routing** **Example:** `Device(l2vpn-config)# pseudowire routing`	(Optional) Enters Layer 2 pseudowire routing configuration mode.
Step 13	**switching-point vcid** *minimum-vcid-value maximum-vcid-value* **Example:** `Device(config-l2_pw_rtg)# switching-point vcid 200 3500`	(Optional) Configures a switching point and specifies a virtual circuit (VC) ID range. **Note** With the L2VPN VPLS Inter-AS Option B feature, VC IDs in the VC ID range of 1001 to 2147483647 are reserved for switching pseudowires. This command allows you to change this range if, for example, an existing xconnect VC is using one of the reserved VC IDs.
Step 14	**exit** **Example:** `Device(config-l2_pw_rtg)# exit`	Exits Layer 2 pseudowire routing configuration mode.
Step 15	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

What to Do Next

Repeat the steps in the Enabling L2VPN VPLS Inter-AS Option B on the ASBR, on page 360 at each ASBR in the autonomous system. Then proceed to the Enabling L2VPN VPLS Inter-AS Option B on the Provider Edge (PE) Router, on page 365.

Enabling L2VPN VPLS Inter-AS Option B on the Provider Edge (PE) Router

To enable the L2VPN VPLS Inter-AS Option B on the PE router, complete the following steps on each PE in the autonomous system.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 pseudowire routing**
4. **terminating-pe tie-breaker**
5. exit
6. **mpls ldp discovery targeted-hello accept**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**l2 pseudowire routing** **Example:** `Router(config))# l2 pseudowire routing`	Enters Layer 2 pseudowire routing configuration mode.
Step 4	**terminating-pe tie-breaker** **Example:** `Router(config-l2_pw_rtg)# terminating-pe tie-breaker`	Negotiates the behavior mode (either active or passive) for a terminating provider edge (TPE) route.
Step 5	exit **Example:** `Router(config-l2_pw_rtg)# exit`	Returns to global configuration mode.
Step 6	**mpls ldp discovery targeted-hello accept** **Example:** `Router(config)# mpls ldp discovery targeted-hello accept`	Configures the routers from which LDP sessions will be accepted. • With the **targeted-hello accept** keywords, LDP sessions from *any* router will be accepted.

	Command or Action	Purpose
		• For the other keyword choices available for this command, see the *Cisco IOS Multiprotocol Label Switching Command Reference* .
Step 7	**end** **Example:** `Router(config)# end`	Exits global configuration mode.

What to Do Next

Repeat the steps in the Enabling L2VPN VPLS Inter-AS Option B on the Provider Edge (PE) Router, on page 365 at each PE in the autonomous system. Then proceed to the Verifying the L2VPN VPLS Inter-AS Option B Configuration, on page 367.

Enabling L2VPN VPLS Inter-AS Option B on the Provider Edge (PE) Router using the commands associated with the L2VPN Protocol-Based CLIs feature

To enable the L2VPN VPLS Inter-AS Option B on the PE router, perform this task on each PE in the autonomous system.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn**
4. **pseudowire routing**
5. **terminating-pe tie-breaker**
6. **end**
7. **mpls ldp discovery targeted-hello accept**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**l2vpn** **Example:** `Device(config)# l2vpn`	(Optional) Enters Layer 2 VPN configuration mode.
Step 4	**pseudowire routing** **Example:** `Device(l2vpn-config)# pseudowire routing`	(Optional) Enters Layer 2 pseudowire routing configuration mode.
Step 5	**terminating-pe tie-breaker** **Example:** `Device(config-l2_pw_rtg)# terminating-pe tie-breaker`	Negotiates the behavior mode (either active or passive) for a terminating provider edge (TPE) route.
Step 6	**end** **Example:** `Device(config-l2_pw_rtg)# exit`	Returns to global configuration mode.
Step 7	**mpls ldp discovery targeted-hello accept** **Example:** `Device(config)# mpls ldp discovery targeted-hello accept`	Configures the routers from which LDP sessions will be accepted. • With the **targeted-hello accept** keywords, LDP sessions from *any* router will be accepted. • For the other keyword choices available for this command, see the *Cisco IOS Multiprotocol Label Switching Command Reference*.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

What to Do Next

Repeat the steps in the Enabling L2VPN VPLS Inter-AS Option B on the Provider Edge (PE) Router, on page 365 at each PE in the autonomous system. Then proceed to the Verifying the L2VPN VPLS Inter-AS Option B Configuration, on page 367.

Verifying the L2VPN VPLS Inter-AS Option B Configuration

To verify the L2VPN VPLS Inter-AS Option B configuration, use one or more of the following commands at any router.

SUMMARY STEPS

1. **enable**
2. **show xconnect rib detail**
3. **show mpls l2transport vc** [**detail**] [**pwid** *pw-identifier*] [**vpls-id** *vpls-identifier*] [**stitch** *endpoint endpoint*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show xconnect rib detail** **Example:** `Router# show xconnect rib detail`	(Optional) Displays the information about the pseudowire Routing Information Base (RIB).
Step 3	**show mpls l2transport vc** [**detail**] [**pwid** *pw-identifier*] [**vpls-id** *vpls-identifier*] [**stitch** *endpoint endpoint*] **Example:** `Router# show mpls l2transport vc`	(Optional) Displays the information about Multiprotocol Label Switching (MPLS) Any Transport over ATM (AToM) VCs and static pseudowires that have been enabled to route Layer 2 packets on a router. • Use the optional keywords and arguments, as applicable.
Step 4	**end** **Example:** `Router# end`	Exits privileged EXEC mode.

Verifying the L2VPN VPLS Inter-AS Option B Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

To verify the L2VPN VPLS Inter-AS Option B configuration, use one or more of the following commands on any router.

SUMMARY STEPS

1. **enable**
2. **show l2vpn rib detail**
3. **show l2vpn atom vc** [**pwid** *pw-identifier*] [**vpls-id** *vpls-identifier*] [**stitch** *endpoint endpoint*][**detail**]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show l2vpn rib detail** **Example:** `Device# show l2vpn rib detail`	(Optional) Displays the information about the pseudowire Routing Information Base (RIB).
Step 3	**show l2vpn atom vc** [**pwid** *pw-identifier*] [**vpls-id** *vpls-identifier*] [**stitch** *endpoint endpoint*][**detail**] **Example:** `Device# show l2vpn atom vc`	(Optional) Displays the information about Multiprotocol Label Switching (MPLS) Any Transport over ATM (AToM) VCs and static pseudowires that have been enabled to route Layer 2 packets on a router. • Use the optional keywords and arguments, as applicable.
Step 4	**end** **Example:** `Device# end`	Exits privileged EXEC mode.

Configuration Examples for L2VPN VPLS Inter-AS Option B

Example Modifying the VPLS Autodiscovery Settings for Use with L2VPN VPLS Inter-AS Option B

In the following example, the VPLS Autodiscovery: BGP Based feature is modified for use with the L2VPN VPLS Inter-AS Option B feature:

```
Router> enable

Router# configure terminal

Router(config)# l2 vfi vpls1 autodiscovery

Router(config-vfi)# vpn id 10

Router(config-vfi)# vpls-id 5:300

Router(config-vfi)# route-target 600:2222
```

```
Router(config-vfi)# exit
```

Example: Modifying the VPLS Autodiscovery Settings for Use with L2VPN VPLS Inter-AS Option B using the commands associated with the L2VPN Protocol-Based CLIs feature

In the following example, the VPLS Autodiscovery: BGP Based feature is modified for use with the L2VPN VPLS Inter-AS Option B feature:

```
Device# enable
Device# configure terminal
Device(config)# l2vpn vfi context vpls1
Device(config-vfi)# vpn id id
Device(config-vfi)# autodiscovery bgp signaling ldp
Device(config-vfi)# vpls-id 5:300
Device(config-vfi)# route-target 600:2222
Device(config-vfi)# exit
```

Example Enabling L2VPN VPLS Inter-AS Option B on the ASBR

In the following example, the L2VPN VPLS Inter-AS Option B feature has been configured on one ASBR:

```
Router> enable

Router# configure terminal

Router(config)# router bgp 1

Router(config-router)# neighbor 10.10.0.1 next-hop-self

Router(config-router)# address-family l2vpn vpls

Router(config-router-af)# no bgp default route-target filter

Router(config-router-af)# exit

Router(config-router)# exit

Router(config)# mpls ldp discovery targeted-hello accept

Router(config)# end
```

Example Enabling L2VPN VPLS Inter-AS Option B on the PE Router

In the following example, the L2VPN VPLS Inter-AS Option B feature is configured on a PE router. The PE is also a TPE.

```
Router> enable

Router# configure terminal

Router(config))# l2 pseudowire routing

Router(config-l2_pw_rtg)# terminating-pe tie-breaker

Router(config-l2_pw_rtg)# exit

Router(config)# mpls ldp discovery targeted-hello accept

Router(config)# end
```

Example Enabling L2VPN VPLS Inter-AS Option B on the PE Device using the commands associated with the L2VPN Protocol-Based CLIs feature

In the following example, the L2VPN VPLS Inter-AS Option B feature is configured on a provider edge (PE) router. The PE is also a terminating provider edge (TPE).

```
Device> enable
Device# configure terminal
Device(config)# l2vpn
Device(l2vpn-config)#  pseudowire routing
Device(config-l2_pw_rtg)# terminating-pe tie-breaker
Device(config-l2_pw_rtg)# exit
Device(config)# mpls ldp discovery targeted-hello accept
Device(config)# end
```

Example Verifying the L2VPN VPLS Inter-AS Option B Configuration

The output of the **show xconnect rib detail**command can be used to verify the L2VPN VPLS Inter-AS Option B configuration.

The following is sample output from the **show xconnect rib detail** command when used in an ASBR configuration. On an ASBR, the **show xconnect rib detail** command displays the Layer 2 VPN BGP Network Layer Reachability Information (NLRI) received from the BGP peers. The display also shows the signaling messages received from the targeted LDP sessions for a given TAII.

```
Router# show xconnect rib detail
Local Router ID: 10.1.1.3
VPLS-ID: 1:1, Target ID: 10.1.1.1
  Next-Hop: 10.1.1.1
  Hello-Source: 10.1.1.3
  Route-Target: 2:2
```

```
  Incoming RD: 10.0.0.0:1
  Forwarder:
  Origin: BGP
  Provisioned: Yes
  SAII: 10.0.0.1, LDP Peer Id: 10.255.255.255, VC Id: 1001 ***
  SAII: 10.1.0.1, LDP Peer Id: 10.255.255.255, VC Id: 1002 ***
```

After the passive TPE router receives the BGP information (and before the passive TPE router receives the LDP label), the peer information will be displayed in the output of the **show xconnect rib** command. The peer information will not be displayed in the **show mpls l2transport vc**command because the VFI AToM xconnect has not yet been provisioned.

Therefore, for passive TPEs, the entry “Passive : Yes” is added to the output of the **show xconnect rib detail** command. In addition, the entry “Provisioned: Yes” is displayed after the neighbor xconnect is successfully created (without any retry attempts).

In the sample output, the two lines beginning with “SAII” show that this ASBR is stitching two provider PE routers (10.0.0.1 and 10.1.0.1) to the TAII 10.1.1.1.

Example Verifying the L2VPN VPLS Inter-AS Option B Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

The output of the **show l2vpn rib detail** command can be used to verify the L2VPN VPLS Inter-AS Option B configuration.

The following is sample output from the **show l2vpn rib detail** command when used in an autonomous system boundary router (ASBR) configuration. On an ASBR, the **show l2vpn rib detail** command displays the Layer 2 VPN BGP Network Layer Reachability Information (NLRI) received from the BGP peers. The display also shows the signaling messages received from the targeted label distribution protocol (LDP) sessions for a given TAII.

```
Device# show l2vpn rib detail
Local Router ID: 10.1.1.3
VPLS-ID: 1:1, Target ID: 10.1.1.1
  Next-Hop: 10.1.1.1
  Hello-Source: 10.1.1.3
  Route-Target: 2:2
  Incoming RD: 10.0.0.0:1
  Forwarder:
  Origin: BGP
  Provisioned: Yes
  SAII: 10.0.0.1, LDP Peer Id: 10.255.255.255, VC Id: 1001 ***
  SAII: 10.1.0.1, LDP Peer Id: 10.255.255.255, VC Id: 1002 ***
```

After the passive terminating provider edge (TPE) router receives the BGP information (and before the passive TPE router receives the LDP label), the peer information will be displayed in the output of the **show l2vpn rib** command. The peer information will not be displayed in the **show l2vpn atom vc** command because the VFI AToM xconnect has not yet been provisioned.

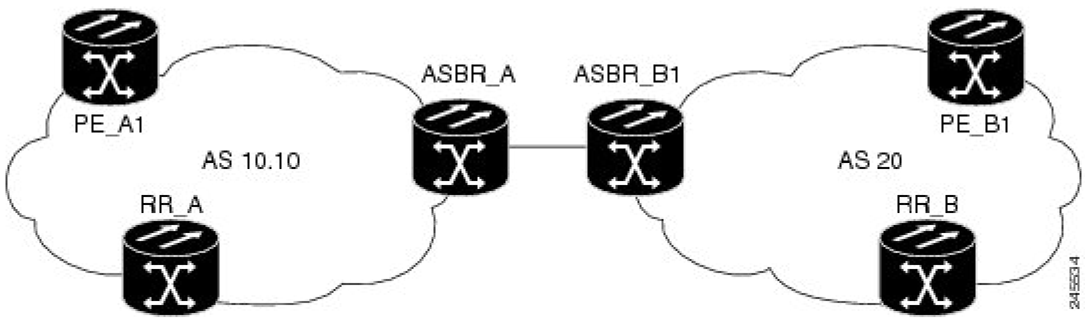
Therefore, for passive TPEs, the entry “Passive : Yes” is added to the output of the **show l2vpn rib detail** command. In addition, the entry “Provisioned: Yes” is displayed after the neighbor xconnect is successfully created (without any retry attempts).

In the sample output, the two lines beginning with “SAII” show that this ASBR is stitching two provider PE routers (10.0.0.1 and 10.1.0.1) to the TAII 10.1.1.1.

Example Sample L2VPN VPLS Inter-AS Option B Configuration

The following is a sample L2VPN VPLS Inter-AS Option B configuration based on the topology shown in the figure below.

Figure 23: L2VPN VPLS Inter-AS Option B Topology Used for Configuration Example



The topology shown in the figure above consists of two PE routers connected across an autonomous system boundary using two ASBRs. Routes are shared within each autonomous system using BGP route reflectors (RRs). (The RRs are included only for the purpose of showing a complete configuration. RRs are not a requirement for the L2VPN Inter-AS Option B configuration.)

The specific configurations for each of the elements in this topology are shown below. The text in bold indicates the additions needed to the standard VPLS Autodiscovery: BGP Based configuration.

PE_A1 Router

```
mpls ldp discovery targeted-hello accept
mpls label protocol ldp
!
l2 router-id 10.1.1.1
!
l2 pseudowire routing
 terminating-pe tie-breaker
!
l2 vfi vfiA autodiscovery
 vpn id 111
 vpls-id 111:111
 rd 111:111
 route-target 111:111
 no auto-route-target
!
!
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
!
interface GigabitEthernet2/0/9
 description AS-10.10-Backbone-LAN
 ip address 10.100.100.1 255.255.255.0
 mpls ip
!
router ospf 10
 network 10.1.1.1 0.0.0.0 area 0
 network 10.100.100.1 0.0.0.0 area 0
!
router bgp 10.10
```

```
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.3.3.3 remote-as 10.10
 neighbor 10.3.3.3 description RR-AS-10.10
 neighbor 10.3.3.3 update-source Loopback0
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 10.3.3.3 activate
  neighbor 10.3.3.3 send-community extended
 exit-address-family
!
mpls ldp router-id Loopback0
!
```

ASBR_A Router

```
mpls ldp discovery targeted-hello accept
mpls label protocol ldp
!
!
interface Loopback0
 ip address 10.4.4.4 255.255.255.255
!
interface GigabitEthernet1/10
 description AS-10.10-backbone-Lan
 ip address 10.100.100.4 255.255.255.0
 mpls ip
!
interface GigabitEthernet2/0/1
 description B2B-AS-20-ASBR-B1
 ip address 10.12.1.4 255.255.255.0
 mpls ip
!
router ospf 10
 passive-interface GigabitEthernet1/12
 passive-interface GigabitEthernet2/0/1
 passive-interface GigabitEthernet2/0/2
 network 10.4.4.4 0.0.0.0 area 0
 network 10.100.100.4 0.0.0.0 area 0
 network 10.12.0.0 0.0.255.255 area 0
!
router bgp 10.10
 bgp router-id 10.4.4.4
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default route-target filter
 no bgp default ipv4-unicast
 timers bgp 10 30
 neighbor AS20 peer-group
 neighbor AS20 remote-as 20
 neighbor 10.3.3.3 remote-as 10.10
 neighbor 10.3.3.3 update-source Loopback0
 neighbor 10.12.1.6 peer-group AS20
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
```

```
 address-family l2vpn vpls
  neighbor AS20 send-community extended
  neighbor AS20 next-hop-self
  neighbor 10.3.3.3 activate
  neighbor 10.3.3.3 send-community extended
  neighbor 10.3.3.3 next-hop-self
  neighbor 12.12.1.6 activate
 exit-address-family
!
ip route 10.6.6.6 255.255.255.255 10.12.1.6
ip route 10.9.9.9 255.255.255.255 10.12.3.9
!
mpls ldp router-id Loopback0
!
```

RR_A Router

```
interface Loopback0
 ip address 10.3.3.3 255.255.255.255
!
interface Ethernet2/0
 ip address 10.100.100.3 255.255.255.0
 duplex half
!
router ospf 10
 network 10.3.3.3 0.0.0.0 area 0
 network 10.100.100.3 0.0.0.0 area 0
!
router bgp 10.10
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor rr-client peer-group
 neighbor rr-client remote-as 10.10
 neighbor rr-client update-source Loopback0
 neighbor 10.1.1.1 peer-group rr-client
 neighbor 10.4.4.4 peer-group rr-client
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor rr-client send-community extended
  neighbor rr-client route-reflector-client
  neighbor 10.1.1.1 activate
  neighbor 10.4.4.4 activate
 exit-address-family
!
```

PE_B1 Router

```
mpls ldp discovery targeted-hello accept
mpls label protocol ldp
!
l2 router-id 10.5.5.5
l2 pseudowire routing
 terminating-pe tie-breaker
l2 vfi vfiA autodiscovery
 vpn id 111
 vpls-id 111:111
```

```
 rd 111:111
 route-target 111:111
 no auto-route-target
!
interface Loopback0
 ip address 10.5.5.5 255.255.255.255
!
interface GigabitEthernet2/0/7
 description AS20-Backbone-LAN
 ip address 10.100.100.5 255.255.255.0
 mpls ip
!
router ospf 20
 network 10.5.5.5 0.0.0.0 area 0
 network 10.100.100.5 0.0.0.0 area 0
!
router bgp 20
 bgp router-id 10.5.5.5
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.8.8.8 remote-as 20
 neighbor 10.8.8.8 update-source Loopback0
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 10.8.8.8 activate
  neighbor 10.8.8.8 send-community extended
 exit-address-family
!
mpls ldp router-id Loopback0
!
```

ASBR_B1 Router

```
mpls ldp discovery targeted-hello accept
mpls label protocol ldp
!
l2 router-id 10.6.6.6
l2 pseudowire routing
 terminating-pe tie-breaker
!
interface Loopback0
 ip address 10.6.6.6 255.255.255.255
!
interface Ethernet1/3
 description B2B-AS-10.10-ASBR-A
 ip address 10.12.1.6 255.255.255.0
 duplex half
 mpls ip
!
interface Ethernet2/1
 description AS-20-backbone-Lan
 ip address 10.100.100.6 255.255.255.0
 duplex half
 mpls ip
!
router ospf 20
 passive-interface Ethernet1/3
 network 10.12.1.6 0.0.0.0 area 0
```

```
 network 10.6.6.6 0.0.0.0 area 0
 network 10.100.100.6 0.0.0.0 area 0
!
router bgp 20
 bgp router-id 10.6.6.6
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 timers bgp 10 30
 neighbor 10.12.1.4 remote-as 10.10
 neighbor 10.12.1.4 ebgp-multihop 255
 neighbor 10.8.8.8 remote-as 20
 neighbor 10.8.8.8 update-source Loopback0
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  no bgp default route-target filter
  neighbor 10.12.1.4 activate
  neighbor 10.12.1.4 send-community extended
  neighbor 10.12.1.4 next-hop-self
  neighbor 10.8.8.8 activate
  neighbor 10.8.8.8 send-community extended
  neighbor 10.8.8.8 next-hop-self
 exit-address-family
!
```

RR_B Router

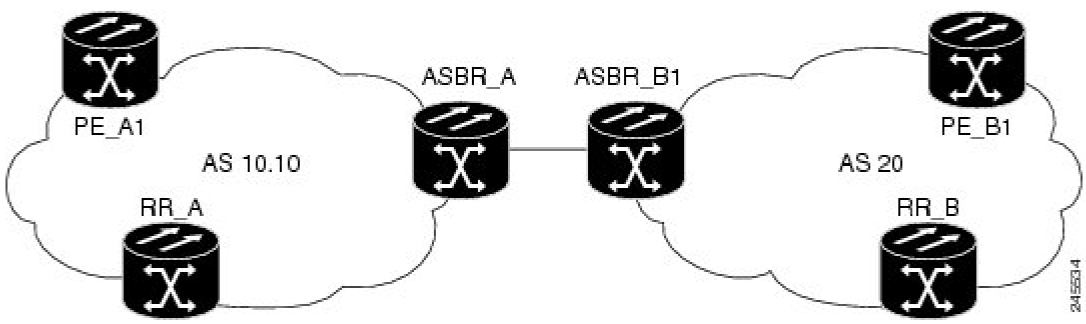
```
interface Loopback0
 ip address 10.8.8.8 255.255.255.255
!
interface Ethernet2/1
 ip address 10.100.100.8 255.255.255.0
 duplex half
!
router ospf 20
 network 10.8.8.8 0.0.0.0 area 0
 network 10.100.100.8 0.0.0.0 area 0
!
router bgp 20
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor rrc peer-group
 neighbor rrc remote-as 20
 neighbor rrc update-source Loopback0
 neighbor 10.5.5.5 peer-group rrc
 neighbor 10.6.6.6 peer-group rrc
 neighbor 10.9.9.9 peer-group rrc
 neighbor 10.9.9.9 shutdown
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor rrc send-community extended
  neighbor rrc route-reflector-client
  neighbor 10.5.5.5 activate
  neighbor 10.6.6.6 activate
  neighbor 10.9.9.9 activate
```

```
 exit-address-family
!
```

Example Sample L2VPN VPLS Inter-AS Option B Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

The example below is a sample L2VPN VPLS Inter-AS Option B configuration based on the topology shown in the following figure.

Figure 24: L2VPN VPLS Inter-AS Option B Topology Used for Configuration Example



The topology shown in the figure above consists of two provider edge (PE) routers connected across an autonomous system boundary using two ASBRs. Routes are shared within each autonomous system using BGP route reflectors (RRs). (The RRs are included only for the purpose of showing a complete configuration. RRs are not a requirement for the L2VPN Inter-AS Option B configuration.)

The specific configurations for each of the elements in this topology are shown below. The commands highlighted in bold indicate the additions needed to the standard VPLS Autodiscovery: BGP Based configuration.

PE_A1 Router

```
mpls ldp discovery targeted-hello accept
mpls label protocol ldp
!
l2vpn
 router-id 10.1.1.1
 pseudowire routing
   terminating-pe tie-breaker
!
l2vpn vfi context vfiA
 vpn id 111
 autodiscovery bgp signaling ldp
 vpls-id 111:111
 rd 111:111
 route-target 111:111
 no auto-route-target
!
!
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
!
interface GigabitEthernet2/0/9
 description AS-10.10-Backbone-LAN
```

```
 ip address 10.100.100.1 255.255.255.0
 mpls ip
!
router ospf 10
 network 10.1.1.1 0.0.0.0 area 0
 network 10.100.100.1 0.0.0.0 area 0
!
router bgp 10.10
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.3.3.3 remote-as 10.10
 neighbor 10.3.3.3 description RR-AS-10.10
 neighbor 10.3.3.3 update-source Loopback0
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 10.3.3.3 activate
  neighbor 10.3.3.3 send-community extended
 exit-address-family
!
mpls ldp router-id Loopback0
!
```

ASBR_A Router

```
mpls ldp discovery targeted-hello accept
mpls label protocol ldp
!
!
interface Loopback0
 ip address 10.4.4.4 255.255.255.255
!
interface GigabitEthernet1/10
 description AS-10.10-backbone-Lan
 ip address 10.100.100.4 255.255.255.0
 mpls ip
!
interface GigabitEthernet2/0/1
 description B2B-AS-20-ASBR-B1
 ip address 10.12.1.4 255.255.255.0
 mpls ip
!
router ospf 10
 passive-interface GigabitEthernet1/12
 passive-interface GigabitEthernet2/0/1
 passive-interface GigabitEthernet2/0/2
 network 10.4.4.4 0.0.0.0 area 0
 network 10.100.100.4 0.0.0.0 area 0
 network 10.12.0.0 0.0.255.255 area 0
!
router bgp 10.10
 bgp router-id 10.4.4.4
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default route-target filter
 no bgp default ipv4-unicast
 timers bgp 10 30
 neighbor AS20 peer-group
 neighbor AS20 remote-as 20
```

```
 neighbor 10.3.3.3 remote-as 10.10
 neighbor 10.3.3.3 update-source Loopback0
 neighbor 10.12.1.6 peer-group AS20
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor AS20 send-community extended
  neighbor AS20 next-hop-self
  neighbor 10.3.3.3 activate
  neighbor 10.3.3.3 send-community extended
  neighbor 10.3.3.3 next-hop-self
  neighbor 12.12.1.6 activate
 exit-address-family
!
ip route 10.6.6.6 255.255.255.255 10.12.1.6
ip route 10.9.9.9 255.255.255.255 10.12.3.9
!
mpls ldp router-id Loopback0
!
```

RR_A Router

```
interface Loopback0
 ip address 10.3.3.3 255.255.255.255
!
interface Ethernet2/0
 ip address 10.100.100.3 255.255.255.0
 duplex half
!
router ospf 10
 network 10.3.3.3 0.0.0.0 area 0
 network 10.100.100.3 0.0.0.0 area 0
!
router bgp 10.10
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor rr-client peer-group
 neighbor rr-client remote-as 10.10
 neighbor rr-client update-source Loopback0
 neighbor 10.1.1.1 peer-group rr-client
 neighbor 10.4.4.4 peer-group rr-client
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor rr-client send-community extended
  neighbor rr-client route-reflector-client
  neighbor 10.1.1.1 activate
  neighbor 10.4.4.4 activate
 exit-address-family
!
```

PE_B1 Router

```
mpls ldp discovery targeted-hello accept
```

```
mpls label protocol ldp
!
l2vpn
 router-id 10.5.5.5
 pseudowire routing
  terminating-pe tie-breaker
l2vpn vfi context vfiA
 vpn id 111
 autodiscovery bgp signaling ldp
 vpls-id 111:111
 rd 111:111
 route-target 111:111
 no auto-route-target
!
interface Loopback0
 ip address 10.5.5.5 255.255.255.255
!
interface GigabitEthernet2/0/7
 description AS20-Backbone-LAN
 ip address 10.100.100.5 255.255.255.0
 mpls ip
!
router ospf 20
 network 10.5.5.5 0.0.0.0 area 0
 network 10.100.100.5 0.0.0.0 area 0
!
router bgp 20
 bgp router-id 10.5.5.5
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.8.8.8 remote-as 20
 neighbor 10.8.8.8 update-source Loopback0
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 10.8.8.8 activate
  neighbor 10.8.8.8 send-community extended
 exit-address-family
!
mpls ldp router-id Loopback0
!
```

ASBR_B1 Router

```
mpls ldp discovery targeted-hello accept
mpls label protocol ldp
!
l2vpn
 router-id 10.6.6.6
 pseudowire routing
  terminating-pe tie-breaker
!
interface Loopback0
 ip address 10.6.6.6 255.255.255.255
!
interface Ethernet1/3
 description B2B-AS-10.10-ASBR-A
 ip address 10.12.1.6 255.255.255.0
 duplex half
```

```
 mpls ip
!
interface Ethernet2/1
 description AS-20-backbone-Lan
 ip address 10.100.100.6 255.255.255.0
 duplex half
 mpls ip
!
router ospf 20
 passive-interface Ethernet1/3
 network 10.12.1.6 0.0.0.0 area 0
 network 10.6.6.6 0.0.0.0 area 0
 network 10.100.100.6 0.0.0.0 area 0
!
router bgp 20
 bgp router-id 10.6.6.6
 bgp asnotation dot
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 timers bgp 10 30
 neighbor 10.12.1.4 remote-as 10.10
 neighbor 10.12.1.4 ebgp-multihop 255
 neighbor 10.8.8.8 remote-as 20
 neighbor 10.8.8.8 update-source Loopback0
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  no bgp default route-target filter
  neighbor 10.12.1.4 activate
  neighbor 10.12.1.4 send-community extended
  neighbor 10.12.1.4 next-hop-self
  neighbor 10.8.8.8 activate
  neighbor 10.8.8.8 send-community extended
  neighbor 10.8.8.8 next-hop-self
 exit-address-family
!
```

RR_B Router

```
interface Loopback0
 ip address 10.8.8.8 255.255.255.255
!
interface Ethernet2/1
 ip address 10.100.100.8 255.255.255.0
 duplex half
!
router ospf 20
 network 10.8.8.8 0.0.0.0 area 0
 network 10.100.100.8 0.0.0.0 area 0
!
router bgp 20
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor rrc peer-group
 neighbor rrc remote-as 20
 neighbor rrc update-source Loopback0
 neighbor 10.5.5.5 peer-group rrc
 neighbor 10.6.6.6 peer-group rrc
 neighbor 10.9.9.9 peer-group rrc
 neighbor 10.9.9.9 shutdown
```

```
 !
 address-family ipv4
  no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor rrc send-community extended
  neighbor rrc route-reflector-client
  neighbor 10.5.5.5 activate
  neighbor 10.6.6.6 activate
  neighbor 10.9.9.9 activate
 exit-address-family
!
```

Additional References for L2VPN VPLS Inter-AS Option B

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference
IP Routing (BGP) commands	Cisco IOS IP Routing: BGP Command Reference
Concepts and tasks related to configuring the VPLS Autodiscovery: BGP Based feature.	*VPLS Autodiscovery BGP Based*
BGP support for the L2VPN address family	*BGP Support for the L2VPN Address Family*
VPLS	"VPLS Overview" section in the *Configuring Multiprotocol Label Switching on the Optical Services Modules* document
L2VPN multisegment pseudowires, MPLS OAM support for L2VPN multisegment pseudowires, MPLS OAM support for L2VPN inter-AS option B	*L2VPN Multisegment Pseudowires*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing standards has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 4360	*BGP Extended Communities Attribute*
RFC 4364	*BGP/MPLS IP Virtual Private Networks (VPNs)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for L2VPN VPLS Inter-AS Option B

Table 21: Feature Information for L2VPN VPLS Inter-AS Option B

Feature Name	Releases	Feature Information
L2VPN VPLS Inter-AS Option B	15.1(1)S Cisco IOS XE Release 3.8S	The L2VPN VPLS Inter-AS Option B feature expands the existing features of VPLS autodiscovery to operate across multiple BGP autonomous systems. Using BGP-based autodiscovery as the underlying framework, the L2VPN VPLS Inter-AS Option B features creates a dynamic multisegmented pseudowire configuration between neighboring ASBRs. The following commands were introduced or modified: **bgp default route-target filter**, **debug xconnect**, **l2 pseudowire routing**, **show ip bgp neighbors**, **show mpls forwarding-table**, **show mpls l2transport vc**, **show xconnect**, **switching-point vcid**, and **terminating-pe tie-breaker**.

Glossary

AGI —Attachment Group Identifier. An identifier common to a group of pseudowires that may be connected.

AII —Attachment individual identifier.

ASBR —Autonomous System Boundary Router.

PE —provider edge router.

NLRI —Network Layer Reachability Information.

SAII —Source Attachment Individual Identifier.

SPE —switching PE.

TAII —Target Attachment Individual Identifier.

TPE —terminating PE.

VFI —virtual forwarding instance. This identifies a group of pseudowires that are associated with a VSI.

VSI —virtual switching instance. This identifies the bridge domain within a single PE. In a single VPLS network, each participating PE has a VSI.

CHAPTER 11

IEEE 802.1Q Tunneling (QinQ) for AToM

This feature allows you to configure IEEE 802.1Q Tunneling (QinQ) for AToM. It also permits the rewriting of QinQ tags for Multiple Protocol Label Switching (MPLS) Layer 2 VPNs (L2VPNs).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for IEEE 802.1Q Tunneling (QinQ) for AToM

The QinQ (short for 802.1Q-in-802.1Q) tunneling and tag rewrite feature is supported on the following line cards:

- 8-port Fast Ethernet line card (ESR-HH-8FE-TX)
- 2-port half-height Gigabit Ethernet line card (ESR-HH-1GE)
- 1-port full-height Gigabit Ethernet line card (ESR-1GE)

Restrictions for IEEE 802.1Q Tunneling (QinQ) for AToM

- Up to a maximum of 447 outer-VLAN IDs and up to 4095 inner VLAN IDs can be supported by this feature.
- Only Unambiguous VLAN tagged Ethernet QinQ interfaces are supported in this release. That is, the Ethernet VLAN QinQ rewrite of both VLAN Tags capability is supported only on Ethernet subinterfaces with a QinQ encapsulation and explicit pair of VLAN IDs defined.

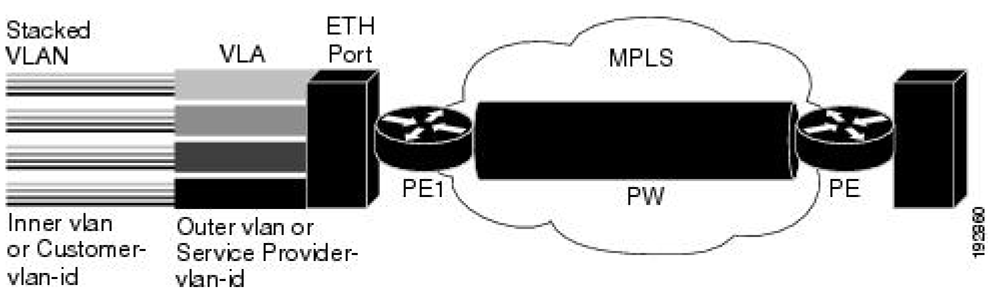
Note Ambiguous inner VLAN IDs are not supported in this release.

Information About IEEE 802.1Q Tunneling (QinQ) for AToM

Ethernet VLAN QinQ AToM

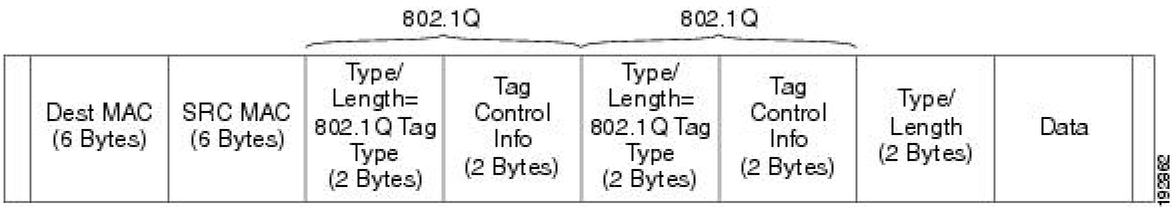
In Metro Ethernet deployment, in which CE routers and PE routers are connected through an Ethernet switched access network, packets that arrive at PE routers can contain up to two IEEE 802.1q VLAN tags (one inner VLAN tag which identifies the customer; and another outer VLAN tag which denotes the customer's service provider). This technique of allowing multiple VLAN tagging on the same Ethernet packet and creating a stack of VLAN IDs is known as QinQ (short for 802.1Q-in-802.1Q). The figure below shows how different edge devices can do L2 switching on the different levels of the VLAN stack.

Figure 25: Ethernet VLAN QinQ



When the outer VLAN tag is the service-delimiting VLAN tag, QinQ packets are processed similar to the ones with one VLAN tag (case previously named Ethernet VLAN Q-in-Q modified, which is already supported in the 12.2(31) SB release). However, when a customer must use a combination of the outer and inner VLAN tags to delimit service for customers, the edge device should be able to choose a unique pseudowire based on a combination of the inner and outer VLAN IDs on the packet shown in the figure below. The customer may want to be able to rewrite both the inner and the outer VLAN IDs on the traffic egress side.

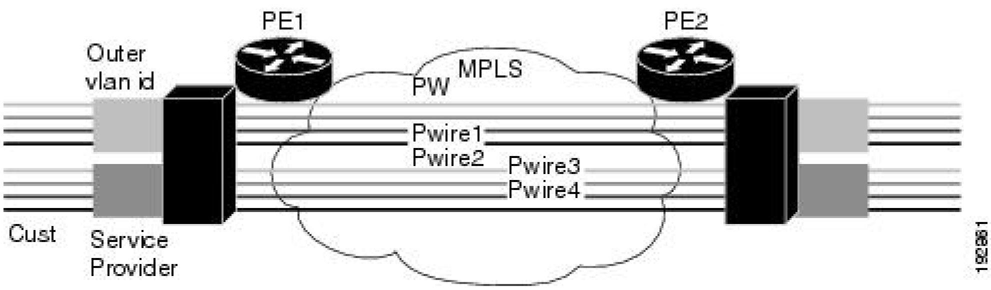
Figure 26: Ethernet VLAN QinQ Header



QinQ Tunneling Based on Inner and Outer VLAN Tags

When handling incoming QinQ Ethernet traffic, the edge router allows a customer to choose a unique pseudowire endpoint to switch the traffic based on the combination of inner and outer VLAN IDs. For example, the figure below shows how a unique pseudowire is selected depending upon the combination of inner (customer edge) and outer (service provider) VLAN IDs. Thus, traffic for different customers can be kept separate.

Figure 27: QinQ Connection



Rewritten Inner and Outer VLAN Tags on QinQ Frames

When managing incoming AToM Ethernet QinQ traffic, the edge router does the following tasks:

1. Strips off the MPLS labels.
2. Allows the customer to rewrite both the inner and outer VLAN IDs before sending the packets to the egress QinQ interface. Note this capability is provided only for AToM like-to-like Ethernet QinQ traffic.

The QinQ AToM feature is a like-to-like interworking case over AToM. This feature requires changes to the microcode to allow it to overwrite two layers of VLAN tags on Ethernet QinQ traffic, transported across AToM pseudowires.

- On the ingress side--The packets preserve their L2 header with the two VLAN tags, and it is sent across the pseudowire with VC type of 4.
- On the egress side--The MPLS label is stripped, and up to two levels of VLAN tags are rewritten per the configuration.

Only Unambiguous VLAN tagged Ethernet QinQ interfaces are supported in this release. The Ethernet VLAN Q-in-Q rewrite of both VLAN Tags capability is supported only on Ethernet subinterfaces with a QinQ encapsulation and explicit pair of VLAN IDs defined.

How to Configure IEEE 802.1Q Tunneling (QinQ) for AToM

This section explains how to configure IEEE 802.1Q Tunneling (QinQ) for AToM and includes the following procedures. While all of the procedures are listed as optional, you must choose one of the first two listed.

Configuring Unambiguous IEEE 802.1Q Tunneling (QinQ) for AToM

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*]
4. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]}
5. **xconnect** *peer-router-id vcid* **encapsulation mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*] **Example:** `Router(config)# interface GigabitEthernet1/0/0.100`	Specifies the Gigabit Ethernet interface and enters interface configuration mode.
Step 4	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]} **Example:** `Router(config-if)# encapsulation dot1q 100 second-dot1q 200`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.
Step 5	**xconnect** *peer-router-id vcid* **encapsulation mpls** **Example:** `Router(config-if)# xconnect 10.0.0.16 410 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.

Configuring Unambiguous IEEE 802.1Q Tunneling (QinQ) for AToM using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*]
4. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]}
5. **interface pseudowire** *number*
6. **encapsulation mpls**
7. **neighbor** *peer-address vcid-value*
8. **exit**
9. **l2vpn xconnect context** *context-name*
10. **member pseudowire** *interface-number*
11. **member gigabitethernet** *interface-number*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*] **Example:** `Router(config)# interface GigabitEthernet1/0/0.100`	Specifies the Gigabit Ethernet interface and enters interface configuration mode.
Step 4	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]} **Example:** `Router(config-if)# encapsulation dot1q 100 second-dot1q 200`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.
Step 5	**interface pseudowire** *number* **Example:**	Specifies the pseudowire interface and enters interface configuration mode.

	Command or Action	Purpose
	`Router(config-if)# interface pseudowire 100`	
Step 6	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 7	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 9	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 10	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 11	**member gigabitethernet** *interface-number* **Example:** `Router(config-xconnect)# member GigabitEthernet1/0/0.100`	Specifies the location of the Gigabit Ethernet member interface.
Step 12	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Configuring Ambiguous IEEE 802.1Q Tunneling (QinQ) for AToM

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*]
4. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]}

5. **xconnect** *peer-router-id vcid encapsulation mpls*
6. **exit**
7. **interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*]
8. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]}
9. **xconnect** *peer-router-id vcid encapsulation mpls*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*] **Example:** `Router(config)# interface GigabitEthernet1/0/0.200`	Specifies the Gigabit Ethernet subinterface and enters interface configuration mode.
Step 4	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]} **Example:** `Router(config-if)# encapsulation dot1q 200 second-dot1q 1000-2000,3000,3500-4000`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.
Step 5	**xconnect** *peer-router-id vcid encapsulation mpls* **Example:** `Router(config-if)# xconnect 10.0.0.16 420 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 7	**interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*] **Example:** `Router(config)# interface GigabitEthernet1/0/0.201`	Specifies the next Gigabit Ethernet interface and enters interface configuration mode.

	Command or Action	Purpose
Step 8	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]} **Example:** `Router(config-if)# encapsulation dot1q 201 second-dot1q any`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.
Step 9	**xconnect** *peer-router-id vcid encapsulation mpls* **Example:** `Router(config-if)# xconnect 10.0.0.16 430 encapsulation mpls`	Creates the VC to transport the Layer 2 packets.

Configuring Ambiguous IEEE 802.1Q Tunneling (QinQ) for AToM using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*]
4. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]}
5. **interface pseudowire** *number*
6. **encapsulation mpls**
7. **neighbor** *peer-address vcid-value*
8. **exit**
9. **interface gigabitethernet** *slot* / *subslot* / *port* **.** [*subinterface*]
10. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]}
11. **interface pseudowire** *number*
12. **encapsulation mpls**
13. **neighbor** *peer-address vcid-value*
14. **exit**
15. **l2vpn xconnect context** *context-name*
16. **member pseudowire** *interface-number*
17. **member gigabitethernet** *interface-number*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *port* . [*subinterface*] **Example:** `Router(config)# interface GigabitEthernet1/0/0.200`	Specifies the Gigabit Ethernet subinterface and enters interface configuration mode.
Step 4	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]} **Example:** `Router(config-if)# encapsulation dot1q 200 second-dot1q 1000-2000,3000,3500-4000`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.
Step 5	**interface pseudowire** *number* **Example:** `Router(config-if)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 6	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 7	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 9	**interface gigabitethernet** *slot* / *subslot* / *port* . [*subinterface*] **Example:** `Router(config)# interface GigabitEthernet1/0/0.201`	Specifies the next Gigabit Ethernet interface and enters interface configuration mode.

	Command or Action	Purpose
Step 10	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlan-id*]]} **Example:** `Router(config-if)# encapsulation dot1q 201 second-dot1q any`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.
Step 11	**interface pseudowire** *number* **Example:** `Router(config-if)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 12	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 13	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 14	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 15	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 16	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 17	**member gigabitethernet** *interface-number* **Example:** `Router(config-xconnect)# member GigabitEthernet1/0/0.201`	Specifies the location of the Gigabit Ethernet member interface.
Step 18	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.

Verifying the IEEE 802.1Q Tunneling (QinQ) for ATM Configuration

SUMMARY STEPS

1. **enable**
2. **show mpls l2transport vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show mpls l2transport vc** **Example:** `Router# show mpls l2transport vc`	Displays information about Any Transport over MPLS (AToM) virtual circuits (VCs) and static pseudowires that have been enabled to route Layer 2 packets on a router.

Verifying the IEEE 802.1Q Tunneling (QinQ) for ATM Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **show l2vpn atom vc**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show l2vpn atom vc** **Example:** `Device# show l2vpn atom vc`	Displays information about Any Transport over MPLS (AToM) virtual circuits (VCs) and static pseudowires that have been enabled to route Layer 2 packets on a router.

Configuration Examples for IEEE 801.2 Tunneling (QinQ) for ATM

Example Configuring Unambiguous IEEE 802.1Q Tunneling (QinQ) for ATM

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet1/0/0.100
Router(config-if)# encapsulation dot1q 100 second-dot1q 200
Router(config-if)# xconnect 10.0.0.16 410 encapsulation mpls
```

Example Configuring Unambiguous IEEE 802.1Q Tunneling (QinQ) for ATM using the commands associated with the L2VPN Protocol-Based CLIs feature

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet1/0/0.100
Router(config-if)# encapsulation dot1q 100 second-dot1q 200
Router(config-if)# interface pseudowire 100
Router(config-if)# encapsulation mpls
Router(config-if)# neighbor 10.0.0.1 123
Router(config-if)# exit
Router(config)# l2vpn xconnect context A
Router(config-xconnect)# member pseudowire 100
Router(config-xconnect)# member GigabitEthernet1/0/0.100
```

Example Configuring Ambiguous IEEE 802.1Q Tunneling (QinQ) for ATM

The following is an example of an ambiguous IEEE 802.1Q Tunneling (QinQ) for ATM configuration.

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet1/0/0.200
Router(config-if)# encapsulation dot1q 200 second-dot1q 1000-2000,3000,3500-4000
Router(config-if)# xconnect 10.0.0.16 420 encapsulation mpls
Router(config-if)# exit
Router(config)# interface GigabitEthernet1/0/0.201
Router(config-if) encapsulation dot1q 201 second-dot1q any
Router(config-if) xconnect 10.0.0.16 430 encapsulation mpls
```

Example Configuring Ambiguous IEEE 802.1Q Tunneling (QinQ) for ATM using the commands associated with the L2VPN Protocol-Based CLIs feature

The following is an example of an ambiguous IEEE 802.1Q Tunneling (QinQ) for ATM configuration.

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet1/0/0.200
Router(config-if)# encapsulation dot1q 200 second-dot1q 1000-2000,3000,3500-4000
```

```
Router(config-if)# interface pseudowire 100
Router(config-if)# encapsulation mpls
Router(config-if)# neighbor 10.0.0.1 123
Router(config-if)# exit
Router(config)# l2vpn xconnect context A
Router(config-xconnect)# member pseudowire 100
Router(config-xconnect)# member GigabitEthernet1/0/0.200
Router(config-xconnect)# exit
Router(config)# interface GigabitEthernet1/0/0.201
Router(config-if) encapsulation dot1q 201 second-dot1q any
Router(config-if)# interface pseudowire 100
Router(config-if)# encapsulation mpls
Router(config-if)# neighbor 10.0.0.1 123
Router(config-if)# exit
Router(config)# l2vpn xconnect context A
Router(config-xconnect)# member pseudowire 100
Router(config-xconnect)# member GigabitEthernet1/0/0.201
```

Example Verifying the IEEE 802.1Q Tunneling (QinQ) for ATM Configuration

The following is sample output of the **show mpls l2transport vc** command, which is usedto verify the VC set up in EoMPLS QinQ mode.

```
router# show mpls l2transport vc
Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
Gi1/0/0.1      Eth VLAN:100/200           10.1.1.2        1          UP
```

Example Verifying the IEEE 802.1Q Tunneling (QinQ) for ATM Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

The following is sample output of the **show l2vpn atom vc** command, which is used to verify the virtual circuit (VC) set up in EoMPLS QinQ mode.

```
Device# show l2vpn atom vc
Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
Gi1/0/0.1      Eth VLAN:100/200           10.1.1.2        1          UP
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Description of commands associated with MPLS and MPLS applications	*Cisco IOS Multiprotocol Label Switching Command Reference*
AToM and MPLS	Any Transport over MPLS

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IEEE 802.1Q Tunneling (QinQ) for AToM

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 22: Feature Information for IEEE 802.1Q Tunneling (QinQ) for AToM

Feature Name	Releases	Feature Information
IEEE 802.1Q Tunneling (QinQ) for AToM	Cisco IOS XE Release 2.4	This feature allows you to configure IEEE 802.1Q Tunneling (QinQ) for AToM. It also permits the rewriting of QinQ tags for Multiple Protocol Label Switching (MPLS) layer 2 VPNs (L2VPNs). In Cisco IOS XE Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. The following commands were introduced or modified: **interface , encapsulation dot1q second-dot1q , xconnect .**

CHAPTER 12

Configuring the Managed IPv6 Layer 2 Tunnel Protocol Network Server

This document describes how to enable the Managed IPv6 Layer 2 Tunnel Protocol Network Server feature.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring the Managed IPv6 LNS

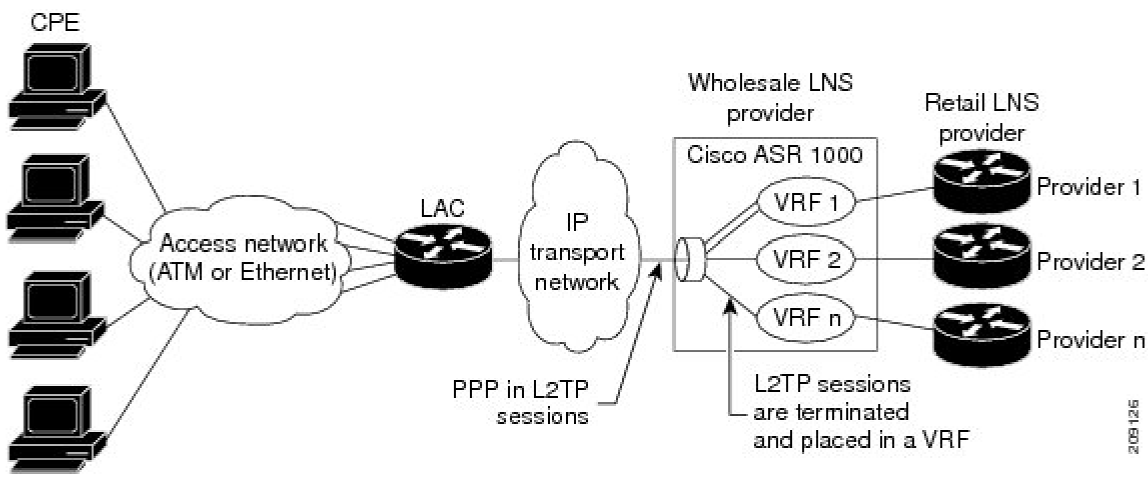
For the router to function as an LNS, you must enable Authentication, Authorization, and Accounting (AAA) on the Layer 2 Tunnel Protocol Network Server (LNS) and the Layer 2 Access Concentrator (LAC), by entering the **aaanew-model** global configuration command. For more information, see the " Authentication, Authorization, and Accounting " chapter in the *Cisco IOS XE Security: Securing User Services Configuration Guide* .

Information About Configuring the Managed IPv6 LNS

L2TP Network Server

The router can function as an LNS. The LNS is a peer to the LAC and sits on one side of an L2TP tunnel. The LNS routes packets to and from the LAC and a destination network. When the router functions as an LNS, you can configure the router to terminate the PPP sessions and route the client IP packets onto the ISP or corporate network toward their final destination (see the figure below). The router can use the Managed IPv6 LNS feature to terminate L2TP sessions from the LAC and place each session into the appropriate IPv6 VRF instance based on the VRF applied to the virtual template interface or alternatively, based on the VRF received for the user through AAA. The router then routes each session within the VRF to the destination network.

Figure 28: Terminating and Forwarding Sessions from the LAC



Tunnel Accounting

The tunnel accounting feature enhances AAA accounting by adding the ability to include tunnel-related statistics in the RADIUS information. Before you can collect tunnel usage information, you must configure the following attributes on the RADIUS server:

- Acct-Tunnel-Connection—Specifies the identifier assigned to the tunnel session. This attribute and the Tunnel-Client-Endpoint and Tunnel-Server-Endpoint attributes provide a way to uniquely identify a tunnel session for auditing purposes.
- Acct-Tunnel-Packets-Lost—Specifies the number of packets lost on a given link.

The table below describes the values for the Acct-Status-Type attribute that support tunnel accounting on the RADIUS server.

Table 23: Acct-Status-Type Values for RADIUS Tunnel Accounting

Acct-Status-Type Values	Value	Description
Tunnel-Link-Reject	14	Marks the rejection of the establishment of a new link in an existing tunnel.
Tunnel-Link-Start	12	Marks the creation of a tunnel link within an L2TP tunnel that carries multiple links.
Tunnel-Link-Stop	13	Marks the destruction of a tunnel link within an L2TP tunnel that carries multiple links.
Tunnel-Reject	11	Marks the rejection of the establishment of a tunnel with another device.
Tunnel-Start	9	Marks the establishment of a tunnel with another device.
Tunnel-Stop	10	Marks the destruction of a tunnel to or from another device.

For more information about the RADIUS tunnel accounting attributes or the Acct-Status-Type values that support RADIUS tunnel accounting, see RFC 2867, RADIUS Accounting Modifications for Tunnel Protocol Support.

For information about RADIUS accounting attributes supported on the Cisco ASR 1000 Series Aggregation Services Routers, see the "RADIUS Attributes" chapter in the Cisco IOS XE Security Configuration Guide: Securing User Services.

For more information on configuring RADIUS, see your RADIUS user documentation.

How to Configure the Managed LNS

Configuring a VRF on the LNS

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf definition** *vrf-name*
4. **rd** *route-distinguisher*
5. **address-family** {**ipv4**|**ipv6**}
6. **route-target** {*import*|*export*|*both*} *route-target-ext-community*
7. **exit-address-family**
8. **address-family** {**ipv4**|**ipv6**}
9. **route-target** {*import*|*export*|*both*} *route-target-ext-community*
10. **end**
11. **show ipv6 route vrf** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Router(config)# vrf definition vrf1`	Configures a VRF routing table and enters VRF configuration mode. • The *vrf-name*argument is the name of the VRF.

Step 4	**rd** *route-distinguisher*
	Example:
	`Router(config-vrf)# rd 100:1`
		Creates routing and forwarding tables for a VRF.
		• The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix. You can enter a route distinguisher in either of these formats:
		  • 16-bit autonomous system number (ASN): your 32-bit number For example, 101:3.
		  • 32-bit IP address: your 16-bit number For example, 192.168.122.15:1.

Step 5	**address-family** {**ipv4**|**ipv6**} **Example:** `Router(config-vrf) address-family ipv6`	Enters VRF address family configuration mode to specify an address family for a VRF. • The **ipv4** keyword specifies an IPv4 address family for a VRF. • The **ipv6** keyword specifies an IPv6 address family for a VRF.
Step 6	**route-target** {*import*|*export*|*both*} *route-target-ext-community* **Example:** `Router(config-vrf-af) route-target both 100:2`	Creates a route-target extended community for a VRF. • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The **both** keyword imports both import and export routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the

	Command or Action	Purpose
		VRF list of import, export, or both (import and export) route-target extended communities.
Step 7	**exit-address-family** **Example:** `Router(config-vrf-af)# exit-address-family`	Exits VRF address family configuration mode and enters VRF configuration mode.
Step 8	**address-family** {**ipv4**|**ipv6**} **Example:** `Router(config-vrf) address-family ipv6`	Enters VRF address family configuration mode to specify an address family for a VRF. • The **ipv4** keyword specifies an IPv4 address family for a VRF. • The **ipv6** keyword specifies an IPv6 address family for a VRF.
Step 9	**route-target** {*import*|*export*|*both*} *route-target-ext-community* **Example:** `Router(config-vrf-af)# route-target both 100:3`	Creates a route-target extended community for a VRF. • The **import** keyword specifies to import routing information from the target VPN extended community. • The **export** keyword specifies to export routing information to the target VPN extended community. • The **both** keyword specifies to import both import and export routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the route-target extended community attributes to the VRF list of import, export, or both (import and export) route-target extended communities. • Enter the **route-target** command one time for each target community.
Step 10	**end** **Example:** `Router(config-vrf-af)# end`	Exits VRF address family configuration mode and returns to privileged EXEC mode.
Step 11	**show ipv6 route vrf** *vrf-name* **Example:** `Router# show ipv6 route vrf vrf1`	Displays the IPv6 routing table associated with a VRF.

Configuring a Virtual Template Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface virtual-template** *number*
4. **vrf forwarding** *name*
5. **ppp authentication chap**
6. **end**
7. **show interfaces virtual-access** *number* **[configuration]**
8. **debug ppp chap**
9. **debug ppp negotiation**
10. **debug ppp negotiation chap**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface virtual-template** *number* **Example:** `Router(config)# interface virtual-template 1`	Creates a virtual template interface and enters interface configuration mode.
Step 4	**vrf forwarding** *name* **Example:** `Router(config-if)# vrf forwarding vpn-1`	(Optional) Maps the virtual template interface to a VRF routing table. **Note** If the VRF assignment is received via the RADIUS server, then this step is not required.
Step 5	**ppp authentication chap** **Example:** `Router(config-if)# ppp authentication chap`	Enables CHAP authentication on the virtual template interface, which is applied to virtual access interfaces (VAI).
Step 6	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 7	**show interfaces virtual-access** *number* **[configuration]** **Example:** `Router# show interfaces virtual-access number [configuration]`	Displays status, traffic data, and configuration information about the VAI you specify.
Step 8	**debug ppp chap** **Example:** `Router# debug ppp chap`	Displays authentication protocol messages for Challenge Authentication Protocol (CHAP) packet exchanges. • This command is useful when a CHAP authentication failure occurs due to a configuration mismatch between devices. Verifying and correcting any username and password mismatch resolves the problem.
Step 9	**debug ppp negotiation** **Example:** `Router# debug ppp negotiation`	Displays information on traffic and exchanges in an internetwork implementing PPP.
Step 10	**debug ppp negotiation chap** **Example:** `Router# debug ppp negotiation chap`	Deciphers a CHAP negotiation problem due to a connectivity problem between a Cisco and non-Cisco device.

Assigning a VRF via the RADIUS Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa authorization configuration** *method-name* **group** *group-name*
4. **ipv6 dhcp pool** *pool-name*
5. **prefix-delegation aaa** [**method-list***method-list*]
6. **dns-server** *ipv6-address*
7. **exit**
8. **interface virtual-template** *number*
9. **ipv6 nd prefix framed-ipv6-prefix**
10. **ipv6 dhcp server** *pool-name* **rapid-commit**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa authorization configuration** *method-name* **group** *group-name* **Example:** `Router(config)# aaa authorization configuration DHCPv6-PD group DHCPv6-PD-RADIUS`	Downloads configuration information from the AAA server using RADIUS.
Step 4	**ipv6 dhcp pool** *pool-name* **Example:** `Router(config)# ipv6 dhcp pool DHCPv6-PD`	Configures a DHCP for IPv6 configuration information pool and enters DHCP for IPv6 pool configuration mode.
Step 5	**prefix-delegation aaa** [**method-list***method-list*] **Example:** `Router(config-dhcpv6)# prefix-delegation aaa method-list DHCPv6-PD`	Specifies that prefixes are to be acquired from AAA servers.
Step 6	**dns-server** *ipv6-address* **Example:** `Router(config-dhcpv6)# dns-server 2001:0DB8:3000:3000::42`	Specifies the Domain Name System (DNS) IPv6 servers available to a DHCP for IPv6 client.
Step 7	**exit** **Example:** `Router(config-dhcpv6)# exit`	Exits DHCP for IPv6 pool configuration mode and enters global configuration mode.
Step 8	**interface virtual-template** *number* **Example:** `Router(config)# interface virtual-template 1`	Creates a virtual template interface that can be configured and applied dynamically in creating VAIs, and enters interface configuration mode.
Step 9	**ipv6 nd prefix framed-ipv6-prefix** **Example:**	Adds the prefix in a received RADIUS framed IPv6 prefix attribute to the interface's neighbor discovery prefix queue.

	Command or Action	Purpose
	`Router(config-if)# ipv6 nd prefix framed-ipv6-prefix`	
Step 10	**ipv6 dhcp server** *pool-name* **rapid-commit** **Example:** `Router(config-if)# ipv6 dhcp server DHCPv6-PD rapid-commit`	Enables DHCPv6 on an interface.
Step 11	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring the LNS to Initiate and Receive L2TP Traffic

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vpdn enable**
4. **vpdn-group** *group-name*
5. **accept-dialin**
6. **protocol l2tp**
7. **virtual-template** *template-number*
8. **exit**
9. **terminate-from hostname** *hostname*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vpdn enable** **Example:**	Enables VPDN networking on the router and informs the router to look for tunnel definitions in a local database and

	Command or Action	Purpose
	`Router(config)# vpdn enable`	on a remote authorization server (home gateway) if one is present.
Step 4	**vpdn-group** *group-name* **Example:** `Router(config)# vpdn-group group1`	Defines a local group name for which you can assign other VPDN variables. • Enters VPDN group configuration mode.
Step 5	**accept-dialin** **Example:** `Router(config-vpdn)# accept-dialin`	Configures the LNS to accept tunneled PPP connections from the LAC and creates an accept-dialin VPDN subgroup. • Enters accept dial-in VPDN subgroup configuration mode.
Step 6	**protocol 12tp** **Example:** `Router(config-vpdn-acc-in)# protocol 12tp`	Specifies the Layer 2 Tunnel Protocol.
Step 7	**virtual-template** *template-number* **Example:** `Router(config-vpdn-acc-in)# virtual-template 1`	Specifies the virtual template to be used to clone VAIs.
Step 8	**exit** **Example:** `Router(config-vpdn-acc-in)# exit`	Returns to VPDN group configuration mode.
Step 9	**terminate-from hostname** *hostname* **Example:** `Router(config-vpdn)# terminate-from hostname lac1-vpn1`	Specifies the hostname of the remote LAC that is required when accepting a VPDN tunnel.
Step 10	**end** **Example:** `Router(config-vpdn)# end`	Exits VPDN configuration mode and returns to privileged EXEC mode.

Limiting the Number of Sessions per Tunnel

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **vpdn-group** *group-name*
4. **accept-dialin**
5. **protocol 12tp**
6. **virtual-template** *template-number*
7. **exit**
8. **terminate-from hostname** *host-name*
9. **session-limit** *limit-number*
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vpdn-group** *group-name* **Example:** `Router(config)# vpdn-group group1`	Defines a local group name for which you can assign other VPDN variables. • Enters VPDN group configuration mode.
Step 4	**accept-dialin** **Example:** `Router(config-vpdn)# accept-dialin`	Configures the LNS to accept tunneled PPP connections from the LAC and creates an accept-dialin VPDN subgroup. • Enters accept dial-in VPDN subgroup configuration mode.
Step 5	**protocol 12tp** **Example:** `Router(config-vpdn-acc-in)# protocol 12tp`	Specifies the Layer 2 Tunnel Protocol.
Step 6	**virtual-template** *template-number* **Example:** `Router(config-vpdn-acc-in)# virtual-template 1`	Specifies the virtual template to be used to clone VAIs.
Step 7	**exit** **Example:** `Router(config-vpdn-acc-in)# exit`	Returns to VPDN group configuration mode.

	Command or Action	Purpose
Step 8	**terminate-from hostname** *host-name* **Example:** `Router(config-vpdn)# terminate-from hostname test_LAC`	Specifies the hostname of the remote LAC that is required when accepting a VPDN tunnel.
Step 9	**session-limit** *limit-number* **Example:** `Router(config-vpdn)# session-limit 100`	Specifies the maximum number of sessions per tunnel.
Step 10	**exit** **Example:** `Router(config-vpdn)# exit`	Exits VPDN configuration mode and returns to privileged EXEC mode.

Configuring RADIUS Attribute Accept or Reject Lists

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa authentication ppp default group** *group-name*
4. **aaa authorization network group group** *group-name*
5. **aaa group server radius** *group-name*
6. **server-private** *ip-address* [**acct-port***port-number*][**timeout***seconds*] [**retransmit***retries*] [**key***string*]
7. **authorization** [**accept**|**reject**] *list-name*
8. **exit**
9. **radius-server attribute list** *listname*
10. **attribute** *value1* [*value2* [*value3...*]]
11. **end**
12. **show accounting**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**aaa authentication ppp default group** *group-name* **Example:** `Router(config)# aaa authentication ppp default group radius_authen1`	Specifies one or more AAA authentication methods for use on serial interfaces running PPP.
Step 4	**aaa authorization network group group** *group-name* **Example:** `Router(config)# aaa authorization network group group radius_authen1`	Sets the parameters that restrict network access to the user.
Step 5	**aaa group server radius** *group-name* **Example:** `Router(config)# aaa group server radius VPDN-Group`	Groups different RADIUS server hosts into distinct lists and distinct methods and enters server group RADIUS configuration mode.
Step 6	**server-private** *ip-address* [**acct-port***port-number*][**timeout***seconds*] [**retransmit***retries*] [**key***string*] **Example:** `Router(config-sg-radius)# server-private 10.1.1.2 acct-port 0 timeout 7 retransmit 3 key cisco1`	Configures the IP address of the private RADIUS server for the group server. • The *ip-address*argument specifies the IP address of the private RADIUS server host. • (Optional) The *port-number*argument specifies the UDP destination port for accounting requests. • (Optional) The *seconds* argument specifies the timeout value (1 to 1000). • (Optional) The *retries*argument specifies the number of times a RADIUS request is re-sent to a server, if that server is not responding or responding slowly. • The *string* argument specifies the authentication and encryption key for all RADIUS communications between the router and the RADIUS server.
Step 7	**authorization** [**accept**|**reject**] *list-name* **Example:** `Router(config-sg-radius)# authorization accept vpn1-autho-list`	Specifies a filter for the attributes that are returned in an Access-Accept packet from the RADIUS server. • The **accept** keyword indicates that all attributes will be rejected except the attributes specified in the *listname* argument. • The **reject** keyword indicates that all attributes will be accepted except for the attributes specified in the *listname* argument and all standard attributes.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Router(config-sg-radius)# exit`	Exits server group RADIUS configuration mode and enters global configuration mode.
Step 9	**radius-server attribute list** *listname* **Example:** `Router(config)# radius-server attribute list vpn1-autho-list`	Defines the list name given to the set of attributes defined using the **attribute** command and enters RADIUS attribute list configuration mode. • Define the *listname* argument to be the same as you defined it in step 7.
Step 10	**attribute** *value1* [*value2* [*value3...*]] **Example:** `Router(config-radius-attrl)# attribute 26,200`	Adds attributes to the configured accept or reject list. • You can use this command multiple times to add attributes to an accept or reject list.
Step 11	**end** **Example:** `Router(config-radius-attrl)# end`	Exits RADIUS attribute list configuration mode and returns to privileged EXEC mode.
Step 12	**show accounting** **Example:** `Router# show accounting`	Displays accounting records for users currently logged in. • Displays active accountable events on the network and helps collect information in the event of a data loss on the accounting server.

Configuring AAA Accounting Using Named Method Lists

Note System accounting does not use named method lists. For system accounting you can define only the default method list. For more information, see the "Configuring Accounting " chapter in the Cisco IOS XE Security Configuration Guide: Securing User Services.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa accounting network** *list-name* **start-stop group radius**
4. **line** [**aux** | **console**| **vty**] [*line-number*]
5. **accounting** {**arap**|**commands***level*|**connection**|**exec**|**resource**} [**default** | *list-name*]
6. **end**
7. **debug aaa accounting**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa accounting network** *list-name* **start-stop group radius** **Example:** `Router(config)# aaa accounting network methodlist start-stop group radius`	Creates an accounting method list and enables accounting.
Step 4	**line** [**aux** | **console**| **vty**] [*line-number*] **Example:** `Router(config)# line console 0`	Enters line configuration mode for the line to which you want to apply the accounting method list.
Step 5	**accounting** {**arap**|**commands***level*|**connection**|**exec**|**resource**} [**default** | *list-name*] **Example:** `Router(config-line)# accounting commands 15 list1`	Applies the accounting method list to a line or a set of lines.
Step 6	**end** **Example:** `Router(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.
Step 7	**debug aaa accounting** **Example:** `Router# debug aaa accounting`	Displays information on accountable events as they occur.

Configuring RADIUS Tunnel Authentication Method Lists on the LNS

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **aaa authorization network** *list-name method1* [*method2...*]
4. **vpdn tunnel authorization network** *lmethod-ist-name method1* [*method2...*]
5. **vpdn tunnel authorization virtual-template** *vtemplate-number*
6. **vpdn tunnel authorization password** *dummy-password*
7. **debug aaa authorization**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa authorization network** *list-name method1* [*method2...*] **Example:** `Router(config)# aaa authorization network mymethodlist group VPDN-Group`	Sets parameters that restrict user access to a network. • The *list-name* argument is a character string used to name the list of authentication methods tried when a user logs in. • **group radius**: Uses the list of all RADIUS servers for authentication. • **group** *group-name*: Uses a subset of RADIUS servers for authentication as defined by the **aaa group server radius** command. • **if-authenticated**: Succeeds if user has been successfully authenticated. • **local**: Uses the local username database for authentication. • **none**: Uses no authentication. **Note** The method list is only for VPDN tunnel authorization and termination, not for domain and Digital Number Identification Service (DNIS) authorization. Therefore, the method list applies only on the tunnel terminator device - the LAC for dialout sessions and the LNS for dialin sessions.
Step 4	**vpdn tunnel authorization network** *lmethod-ist-name method1* [*method2...*] **Example:**	Specifies the AAA method list to use for VPDN remote tunnel hostname-based authorization.

	Command or Action	Purpose
	`Router(config)# vpdn tunnel authorization network mymethodlist`	• If you do not specify a method list (including a default method list) by using the **vpdn tunnel authorization network** command, local authorization occurs by using the local VPDN group configuration.
Step 5	**vpdn tunnel authorization virtual-template** *vtemplate-number* **Example:** `Router(config)# vpdn tunnel authorization virtual-template 10`	Specifies the default virtual template interface used to clone a VAI. • If you do not specify a virtual template interface in the local VPDN group configuration or in a remote RADIUS configuration, then the default virtual template interface is used.
Step 6	**vpdn tunnel authorization password** *dummy-password* **Example:** `Router(config)# vpdn tunnel authorization password mypassword`	Specifies the password to use for the RADIUS authorization request to retrieve the tunnel configuration based on the remote tunnel hostname.
Step 7	**debug aaa authorization** **Example:** `Router# debug aaa authorization`	Displays information on AAA authorization.

Configuring the LNS for RADIUS Tunnel Authentication

Perform the following tasks to configure LNS for RADIUS Tunnel Authentication:

Note Cisco ASR 1000 Series Aggregation Services Routers supports L2TP tunnel authorization. However, RADIUS does not provide attributes for such parameter values as L2TP tunnel timeouts, L2TP tunnel hello intervals, and L2TP tunnel receive window size. When the Cisco ASR 1000 Series Aggregation Services Router does not receive a RADIUS attribute for a parameter, the router uses the default value.

Configuring RADIUS Tunnel Authentication Method Lists on the LNS

To configure method lists on the LNS for RADIUS tunnel authentication, perform the following task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa authorization network** *list-name method1* [*method2...*]
4. **vpdn tunnel authorization network** *method- list-name*
5. **vpdn tunnel authorization virtual-template** *vtemplate-number*

6. **vpdn tunnel authorization password** *dummy-password*
7. **end**
8. **debug aaa authorization**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa authorization network** *list-name method1* [*method2...*] **Example:** `Router(config)# aaa authorization network mymethodlist group VPDN-Group`	Sets parameters that restrict user access to a network • The*list-name*argument is a character string used to name the list of authentication methods tried when a user logs in. • **groupradius**—Uses the list of all RADIUS servers for authentication. • **group***group-name*—Uses a subset of RADIUS servers for authentication as defined by the **aaagroupserverradius** command. • **if-authenticated**—Succeeds if user has been successfully authenticated. • **local**—Uses the local username database for authentication. • **none**—Uses no authentication. **Note** The method list is only for VPDN tunnel authorization and termination, not for domain and Digital Number Identification Service (DNIS) authorization. Therefore, the method list applies only on the tunnel terminator device—the LAC for dialout sessions and the LNS for dialin sessions.
Step 4	**vpdn tunnel authorization network** *method- list-name* **Example:** `Router(config)# vpdn tunnel authorization network mymethodlist`	Specifies the AAA method list to use for VPDN remote tunnel hostname-based authorization. • If you do not specify a method list (including a default method list) by using the **vpdntunnelauthorizationnetwork** command, local authorization occurs by using the local VPDN group configuration.

	Command or Action	Purpose
Step 5	**vpdn tunnel authorization virtual-template** *vtemplate-number* **Example:** `Router(config)# vpdn tunnel authorization virtual-template 10`	Specifies the default virtual template interface used to clone a VAI. • If you do not specify a virtual template interface in the local VPDN group configuration or in a remote RADIUS configuration, then the default virtual template interface is used. **Note** The **vpdntunnelauthorizationvirtual-template** command is applicable only on the LNS.
Step 6	**vpdn tunnel authorization password** *dummy-password* **Example:** `Router(config)# vpdn tunnel authorization password mypassword`	Specifies the password to use for the RADIUS authorization request to retrieve the tunnel configuration based on the remote tunnel hostname. • By default, the password is cisco, but you can configure a different password. **Note** The **vpdntunnelauthorizationpassword** command is applicable on both the LAC and LNS.
Step 7	**end** **Example:** `Router(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 8	**debug aaa authorization** **Example:** `Router# debug aaa authorization`	Displays information on AAA authorization.

Configuring AAA Authentication Methods

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa new-model**
4. Configure RADIUS security protocol parameters. For more information about RADIUS, see the “Configuring RADIUS” chapter in the Cisco IOS XE Security Configuration Guide: Securing User Services .
5. **aaa authentication**
6. Apply the authentication method lists to an interface, a line, or a set of lines as required. For more information about authentication method lists, see the “Configuring Authentication” chapter in the Cisco IOS XE Security Configuration Guide: Securing User Services .
7. **end**

DETAILED STEPS

Step 1 **enable**

Step 2 **configure terminal**

Step 3 **aaa new-model**

Enter this command in global configuration mode to enable AAA.

Step 4 Configure RADIUS security protocol parameters. For more information about RADIUS, see the "Configuring RADIUS" chapter in the Cisco IOS XE Security Configuration Guide: Securing User Services .

Step 5 **aaa authentication**

Enter this command to define the authentication method lists.

Step 6 Apply the authentication method lists to an interface, a line, or a set of lines as required. For more information about authentication method lists, see the "Configuring Authentication" chapter in the Cisco IOS XE Security Configuration Guide: Securing User Services .

Step 7 **end**

Configuration Examples for the Managed IPv6 Layer 2 Tunnel Protocol Network Server

Example Managed IPv6 LNS Configuration

The following example shows how to configure Managed IPv6 LNS features on the router. In this example, the router terminates the tunnel from the LAC and associates the VRFs with the interfaces and the virtual template interfaces. This configuration also shows how to configure RADIUS attribute screening and AAA accounting for the VRFs.

```
!
!
vrf definition Mgmt-intf
 !
 address-family ipv4
 exit-address-family
 !
 address-family ipv6
 exit-address-family
!
vrf definition user_vrf1
 rd 1:1
 route-target export 1:1
 route-target import 1:1
 !
 address-family ipv6
 exit-address-family
!
logging buffered 10000000
enable password lab
```

```
!
aaa new-model
!
!
aaa group server radius radius_authen1
 server-private 10.1.1.2 acct-port 0 timeout 7 retransmit 3 key cisco1
 ip radius source-interface Loopback20000
!
aaa authentication login default none
aaa authentication ppp default group radius_authen1
aaa authorization network default group radius_authen1
aaa authorization configuration DHCPv6-PD group radius_authen1
!
!
!
!
!
aaa session-id common
aaa policy interface-config allow-subinterface
ppp hold-queue 80000
clock timezone EST -5 0
ip source-route
no ip gratuitous-arps
!
!
!
!
!
!
!
no ip domain lookup
ip host mcp-matrix 10.0.0.2
ip host mcp-sun-2 10.0.0.2
!
!
ipv6 unicast-routing
ipv6 dhcp binding track ppp
ipv6 dhcp pool ipv6_dhcp_pool1
 prefix-delegation aaa method-list DHCPv6-PD
!
!
!
!
multilink bundle-name authenticated
vpdn enable
!
vpdn-group VPDN_LNS1
 accept-dialin
  protocol l2tp
  virtual-template 1
 terminate-from hostname test_LAC1
 source-ip 10.0.0.2
 local name test_LNS1
 l2tp tunnel password 0 tunnel1
 l2tp tunnel receive-window 100
 l2tp tunnel timeout no-session 30
 l2tp tunnel retransmit retries 7
 l2tp tunnel retransmit timeout min 2
!
!
no virtual-template snmp
!
!
!
```

```
!
!
!
!
!
!
username asifp1@test1 password 0 hello1
!
redundancy
 notification-timer 30000
 mode none
!
!
!
!
!
ip tftp source-interface GigabitEthernet 0
!
!
!
!
!
!
!
!
interface Loopback1
 no ip address
!
interface Loopback20000
 ip address 209.165.202.131 255.255.255.224
!
interface GigabitEthernet1/1/0
 mac-address 8888.8888.8888
 no ip address
 load-interval 30
 negotiation auto
 hold-queue 4096 in
 hold-queue 4096 out
!
interface GigabitEthernet1/1/0.1
 encapsulation dot1Q 3
 ip address 209.165.202.132 255.255.255.224
!
interface GigabitEthernet1/1/1
 mac-address 4444.4444.4444
 no ip address
 load-interval 30
 no negotiation auto
 hold-queue 4096 in
 hold-queue 4096 out
!
interface GigabitEthernet1/1/1.1
 vrf forwarding user_vrf1
 encapsulation dot1Q 2
 ipv6 address 12::1/72
!
interface GigabitEthernet1/1/2
 no ip address
 negotiation auto
!
interface GigabitEthernet1/1/3
 no ip address
 negotiation auto
!
```

```
interface GigabitEthernet1/1/4
 no ip address
 negotiation auto
!
interface GigabitEthernet1/1/5
 no ip address
 negotiation auto
!
interface GigabitEthernet1/1/6
 no ip address
 negotiation auto
!
interface GigabitEthernet1/1/7
 description Connected to RADIUS
 ip address 209.165.201.1 255.255.255.224
 negotiation auto
!
interface GigabitEthernet1/3/0
 no ip address
 media-type sfp
 negotiation auto
!
interface GigabitEthernet1/3/1
 no ip address
 media-type sfp
 negotiation auto
!
interface GigabitEthernet 0
 vrf forwarding Mgmt-intf
 ip address 209.165.201.1 255.255.255.224
 negotiation auto
!
interface Virtual-Template 1
 no ip address
 no logging event link-status
 ipv6 dhcp server ipv6_dhcp_pool1 rapid-commit
 keepalive 30
 ppp mtu adaptive
 ppp authentication pap
!
ip default-gateway 10.1.0.5
ip forward-protocol nd
!
no ip http server
no ip http secure-server
ip route vrf Mgmt-intf 209.165.201.1 255.255.255.254 172.16.1.1
ip route vrf Mgmt-intf 209.165.201.29 255.255.255.224 172.16.0.1
!
ip radius source-interface GigabitEthernet1/1/7
logging esm config
cdp run
ipv6 route vrf user_vrf1 ::/0 12::2
!
ipv6 neighbor 12::2 GigabitEthernet1/1/1.1 2222.2222.2222
!
!
!
control-plane
!
call admission limit 90
!
!
!
alias exec call show caller summ
```

```
alias exec caller show caller summ
alias exec palt show plat
alias exec plat show platform
alias exec evsi sho plat hard cpp act feat ess stat
!
line con 0
 exec-timeout 0 0
 stopbits 1
line vty 0 4
 exec-timeout 0 0
 password password1
!
exception data-corruption buffer truncate
end
```

Example LNS Tunnel Accounting Configuration

The following example shows how to configure the LNS to send tunnel accounting records to the RADIUS server:

```
aaa new-model
!
!
aaa accounting network m1 start-stop group radius
aaa accounting network m2 stop-only group radius
aaa session-id common
enable secret 5 $1$ftf.$wE6Q5Yv6hmQiwL9pizPCg1
!
username ENT_LNS password 0 tunnelpass
username user1@example.com password 0 lab
username user2@example.com password 0 lab
spe 1/0 1/7
firmware location system:/ucode/mica_port_firmware
spe 2/0 2/9
firmware location system:/ucode/mica_port_firmware
!
!
resource-pool disable
clock timezone est 2
!
ip subnet-zero
no ip domain-lookup
ip host CALLGEN-SECURITY-V2 10.24.80.28 10.47.0.0
ip host dirt 172.16.1.129
!
vpdn enable
vpdn tunnel accounting network m1
vpdn session accounting network m1
!
vpdn-group 1
accept-dialin
protocol l2tp
virtual-template 1
terminate-from hostname ISP_LAC
local name ENT_LNS
!
isdn switch-type primary-5ess
!
!
fax interface-type modem
mta receive maximum-recipients 0
!
```

```
interface Loopback 0
ip address 172.16.0.101 255.255.255.0
!
interface Loopback 1
ip address 192.168.0.101 255.255.255.0
!
interface Ethernet 0
ip address 10.1.26.71 255.255.255.0
no ip mroute-cache
no cdp enable
!
interface virtual-template 1
ip unnumbered Loopback 0
peer default ip address pool vpdn-pool1
ppp authentication chap
!
interface virtual-template 2
ip unnumbered Loopback1
peer default ip address pool vpdn-pool2
ppp authentication chap
!
interface fastethernet 0
no ip address
no ip mroute-cache
shutdown
duplex auto
speed auto
no cdp enable
!
ip local pool vpdn-pool1 172.16.5.1 172.16.128.100
ip local pool vpdn-pool2 10.0.0.1 10.0.0.100
ip default-gateway 10.1.26.254
ip classless
ip route 0.0.0.0 0.0.0.0 10.1.26.254
ip route 192.168.1.2 255.255.255.255 10.1.26.254
no ip http server
ip pim bidir-enable
!
!
dialer-list 1 protocol ip permit
no cdp run
!
!
radius-server host 172.16.192.80 auth-port 1645 acct-port 1646 key rad123
radius-server retransmit 3
call rsvp-sync
end
```

Note For additional accounting examples, see the “Configuring Accounting” chapter in the Cisco IOS XE Security: Secure Services Configuration Guide .

Example Verifying the User Profile on the RADIUS Server

The following is an example user profile on the RADIUS server. The Cisco ASR 1000 Series Aggregation Services Routers retrieves the information in the user profile from the RADIUS server.

```
Radius Profile "user1"
Auth-Type = Local, User-Password = "pwd"
```

```
User-Service-Type = Framed-User
Framed-Protocol = PPP
cisco-avpair = "lcp:interface-config=vrf forwarding VRF01"
cisco-avpair = "lcp:interface-config=ipv6 unnumbered loopback1"
Framed-IPv6-Prefix = "2001:DB8:4567:1234::/64"
Delegated-IPv6-Prefix = "2001:DB8:AAAA::/48"
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Cisco IOS XE MPLS commands	*Cisco IOS MPLS Command Reference*
Authentication, authorization and accounting	Authentication, Authorization, and Accounting (AAA)
Configuring RADIUS	Configuring RADIUS
Configuring accounting	Configuring Accounting
RADIUS attributes	"RADIUS Attributes Overview and RADIUS IETF Attributes" module in the *Cisco IOS XE Security Configuration Guide: Securing User Services*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2867	RADIUS Accounting Modifications for Tunnel Protocol Support

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Configuring Managed IPv6 Layer 2 Tunnel Protocol Network Server

Table 24: Feature Information for Managed IPv6 Layer 2 Tunnel Protocol Network Server

Feature Name	Releases	Feature Information
Managed IPv6 Layer 2 Tunnel Protocol Network Server	Cisco IOS XE Release 3.3S	The Managed IPv6 LNS feature allows the service provider to offer a scalable end-to-end VPN of both IPv4 and IPv6 service to remote users. This feature integrates the Multiprotocol Label Switching (MPLS)-enabled backbone with broadband access capabilities. The following commands were introduced or modified: atm pppatm passive, radius-server attribute list, radius-server key, radius-server retransmit, radius-server vsa send.
Managed IPv6 Layer 2 Tunnel Protocol Network Server - VRF-Lite only	Cisco IOS XE Release 3.3S	The Managed IPv6 LNS feature allows the service provider to offer a scalable end-to-end VPN of both IPv4 and IPv6 service to remote users. This feature integrates the VRF-Lite enabled backbone with broadband access capabilities.
Managed IPv6 Layer 2 Tunnel Protocol Network Server - MPLS VPN	Cisco IOS XE Release 3.7S	The Managed IPv6 LNS feature allows the service provider to offer a scalable end-to-end VPN of both IPv4 and IPv6 service to remote users. This feature integrates the MPLS enabled backbone with broadband access capabilities.

CHAPTER 13

L2VPN Pseudowire Redundancy

The L2VPN Pseudowire Redundancy feature lets you configure your network to detect a failure in the network and reroute the Layer 2 (L2) service to another endpoint that can continue to provide service. This feature provides the ability to recover from a failure either of the remote provider edge (PE) router or of the link between the PE and customer edge (CE) routers.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for L2VPN Pseudowire Redundancy

- This feature module requires that you understand how to configure basic L2 virtual private networks (VPNs).
  - Any Transport over MPLS
  - L2 VPN Interworking
  - Layer 2 Tunneling Protocol Version 3 (L2TPv3)

- The L2VPN Pseudowire Redundancy feature requires that the following mechanisms be in place to enable you to detect a failure in the network:
    - Label-switched paths (LSP) Ping/Traceroute and Any Transport over MPLS Virtual Circuit Connection Verification (AToM VCCV)
    - Local Management Interface (LMI)
    - Operation, Administration, and Maintenance (OAM)

Restrictions for L2VPN Pseudowire Redundancy

- The default Label Distribution Protocol (LDP) session hold-down timer will enable the software to detect failures in about 180 seconds. That time can be configured so that the software can detect failures more quickly. See the **mpls ldp holdtime** command for more information.
- L2VPN Pseudowire Redundancy does not support pseudowire interworking mode with L2TPv3. The connectivity between CEs may be impacted if you have interworking IP configured in the pseudowire class.
- The primary and backup pseudowires must run the same type of transport service. The primary and backup pseudowires must be configured with AToM or L2TPv3.
- The backup peer can only be configured for nonstatic L2TPv3 sessions. The backup L2TPv3 session cannot be static L2TPv3 session. The encapsulation type of primary and backup pseudowire must be the same.
- If you use L2VPN Pseudowire Redundancy with L2VPN Interworking, the interworking method must be the same for the primary and backup pseudowires.
- L2VPN Pseudowire Redundancy does support setting the experimental (EXP) bit on the Multiprotocol Label Switching (MPLS) pseudowire.
- L2VPN Pseudowire Redundancy does not support different pseudowire encapsulation types on the MPLS pseudowire.
- The **mpls l2transport route** command is not supported. Use the **xconnect** command instead.
- The ability to have the backup pseudowire fully operational at the same time that the primary pseudowire is operational is not supported. The backup pseudowire becomes active only after the primary pseudowire fails.
- The AToM VCCV feature is supported only on the active pseudowire.
- More than one backup pseudowire is not supported.

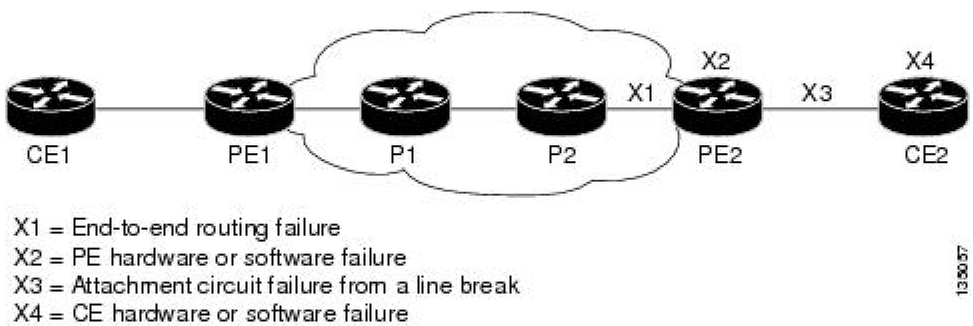
Information About L2VPN Pseudowire Redundancy

Introduction to L2VPN Pseudowire Redundancy

L2VPNs can provide pseudowire resiliency through their routing protocols. When connectivity between end-to-end PE routers fails, an alternative path to the directed LDP session and the user data can take over.

However, there are some parts of the network where this rerouting mechanism does not protect against interruptions in service. The figure below shows those parts of the network that are vulnerable to an interruption in service.

Figure 29: Points of Potential Failure in an L2VPN Network

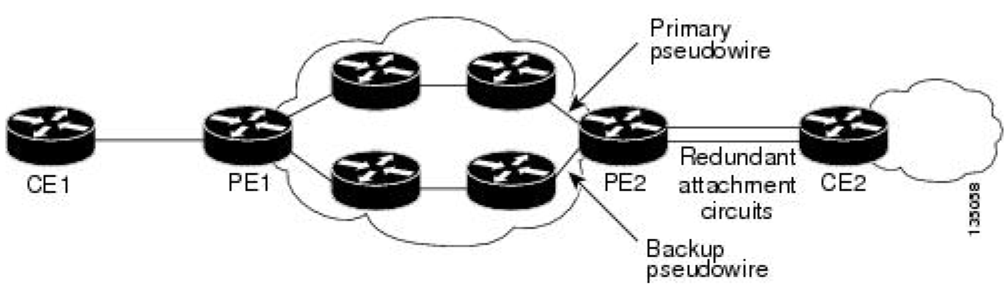


The L2VPN Pseudowire Redundancy feature provides the ability to ensure that the CE2 router in the figure above can always maintain network connectivity, even if one or all the failures in the figure occur.

The L2VPN Pseudowire Redundancy feature enables you to set up backup pseudowires. You can configure the network with redundant pseudowires and redundant network elements, which are shown in the three figures below.

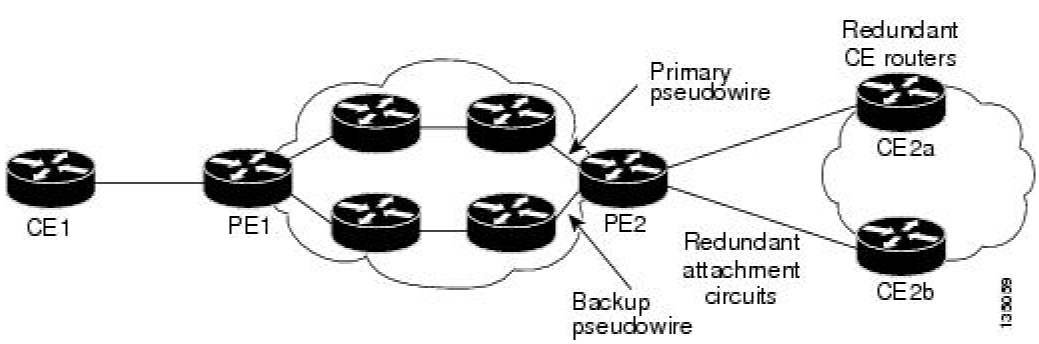
The figure below shows a network with redundant pseudowires and redundant attachment circuits.

Figure 30: L2VPN Network with Redundant PWs and Attachment Circuits



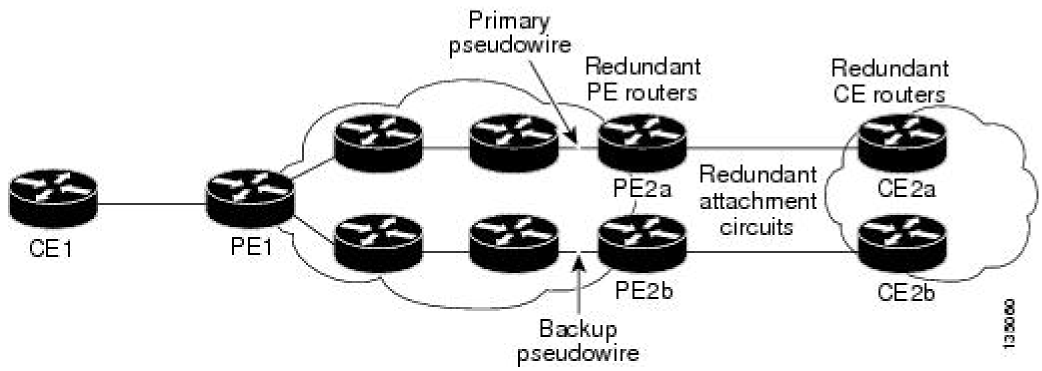
The figure below shows a network with redundant pseudowires, attachment circuits, and CE routers.

Figure 31: L2VPN Network with Redundant PWs, Attachment Circuits, and CE Routers



The figure below shows a network with redundant pseudowires, attachment circuits, CE routers, and PE routers.

Figure 32: L2VPN Network with Redundant PWs, Attachment Circuits, CE Routers, and PE Routers



How to Configure L2VPN Pseudowire Redundancy

The L2VPN Pseudowire Redundancy feature enables you to configure a backup pseudowire in case the primary pseudowire fails. When the primary pseudowire fails, the PE router can switch to the backup pseudowire. You can have the primary pseudowire resume operation after it comes back up.

Configuring the Pseudowire

The successful transmission of the Layer 2 frames between PE routers is due to the configuration of the PE routers. You set up the connection, called a pseudowire, between the routers.

The pseudowire-class configuration group specifies the characteristics of the tunneling mechanism, which are:

- Encapsulation type
- Control protocol
- Payload-specific options

You must specify the **encapsulation mpls**command as part of the pseudowire class for the AToM VCs to work properly. If you omit the **encapsulation mpls**command as part of the **xconnect**command, you receive the following error:

```
% Incomplete command.
```

Perform this task to configure a pseudowire class.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** name
4. **encapsulation mpls**
5. **interworking** {**ethernet** | **ip**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class** name **Example:** `Router(config)# pseudowire-class atom`	Establishes a pseudowire class with a name that you specify. Enters pseudowire class configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation. For AToM, the encapsulation type is **mpls**.
Step 5	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw-class)# interworking ip`	(Optional) Enables the translation between the different Layer 2 encapsulations.

Configuring the Pseudowire using the commands associated with the L2VPN Protocol-Based CLIs feature

The successful transmission of the Layer 2 frames between PE routers is due to the configuration of the PE routers. You set up the connection, called a pseudowire, between the routers.

The pseudowire-class configuration group specifies the characteristics of the tunneling mechanism, which are:

- Encapsulation type
- Control protocol
- Payload-specific options

You must specify the **encapsulation mpls**command as part of the pseudowire class for the AToM VCs to work properly. If you omit the **encapsulation mpls**command as part of the **l2vpn xconnectcontext** command, you receive the following error:

```
% Incomplete command.
```

Perform this task to configure a pseudowire class.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *number*
4. **encapsulation mpls**
5. **neighbor** *peer-address vcid-value*
6. **interworking** {**ethernet** | **ip**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 1`	Establishes an interface pseudowire with a value that you specify. Enters pseudowire configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-pw)# encapsulation mpls`	Specifies the tunneling encapsulation. For AToM, the encapsulation type is **mpls**.
Step 5	**neighbor** *peer-address vcid-value* **Example:** `Router(config-pw)#  neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 6	**interworking** {**ethernet** | **ip**} **Example:** `Router(config-pw)# interworking ip`	(Optional) Enables the translation between the different Layer 2 encapsulations.

Configuring L2VPN Pseudowire Redundancy

Perform this task to configure the L2VPN Pseudowire Redundancy feature.

Before you begin

For each transport type, the **xconnect**command is configured slightly differently. The following configuration steps use Ethernet VLAN over MPLS, which is configured in subinterface configuration mode. See *Any Transport over MPLS* to determine how to configure the **xconnect** command for other transport types.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *interface* **.** *subinterface*
4. **encapsulation dot1q** *vlan-id*
5. **xconnect** *peer-router-id vcid* {**encapsulation mpls**| **pw-class** *pw-class-name}*
6. **backup peer** *peer-router-ip-addr vcid* [**pw-class** *pw-class-name*]
7. **backup delay** *e nable-delay* {*disable-delay* | **never**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *interface* **.** *subinterface* **Example:** `Router(config)# interface gigabitethernet0/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. **Note** Make sure that the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 4	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets. **Note** The subinterfaces between the CE and PE routers that are running Ethernet over MPLS must be in the same subnet.
Step 5	**xconnect** *peer-router-id vcid* {**encapsulation mpls**| **pw-class** *pw-class-name}* **Example:** `Router(config-subif)# xconnect 10.0.0.1 123 pw-class atom`	Binds the attachment circuit to a pseudowire VC and enters xconnect configuration mode. • The syntax for this command is the same as for all other Layer 2 transports.

	Command or Action	Purpose
Step 6	**backup peer** *peer-router-ip-addr vcid* [**pw-class** *pw-class-name*] **Example:** `Router(config-if-xconn)# backup peer 10.0.0.3 125 pw-class atom`	Specifies a redundant peer for the pseudowire VC. The pseudowire class name must match the name that you specified when you created the pseudowire class, but you can use a different pw-class in the **backup peer** command than the name that you used in the primary **xconnect** command.
Step 7	**backup delay** *e nable-delay* {*disable-delay* | **never**} **Example:** `Router(config-if-xconn)# backup delay 5 never`	Specifies how long (in seconds) the backup pseudowire VC should wait to take over after the primary pseudowire VC goes down. The range is from 0 to 180. Specifies how long the primary pseudowire should wait after it becomes active to take over for the backup pseudowire VC. The range is from 0 to 180 seconds. If you specify the **never** keyword, the primary pseudowire VC never takes over for the backup.

Configuring L2VPN Pseudowire Redundancy using the commands associated with the L2VPN Protocol-Based CLIs feature

Perform this task to configure the L2VPN Pseudowire Redundancy feature.

Before you begin

For each transport type, the **l2vpn xconnect context** command is configured slightly differently. The following configuration steps use Ethernet VLAN over MPLS, which is configured in subinterface configuration mode. See *Any Transport over MPLS* to determine how to configure the **l2vpn xconnect context** command for other transport types.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot* / *subslot* / *interface* **.** *subinterface*
4. **encapsulation dot1q** *vlan-id*
5. **end**
6. **interface pseudowire** *number*
7. **source template type pseudowire** *template-name*
8. **neighbor** *peer-address vcid-value*
9. **exit**
10. **l2vpn xconnect context** *context-name*
11. **member pseudowire** *interface-number*
12. **member pseudowire** *interface-number*
13. **member gigabitethernet** *interface-number*
14. **redundancy delay** *enable-delay*{*disable-delay* | **never**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot* / *subslot* / *interface* **.** *subinterface* **Example:** `Device(config)# interface gigabitethernet0/0/0.1`	Specifies the Gigabit Ethernet subinterface and enters subinterface configuration mode. Make sure that the subinterface on the adjoining CE router is on the same VLAN as this PE router.
Step 4	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1q 100`	Enables the subinterface to accept 802.1Q VLAN packets. The subinterfaces between the CE and PE routers that are running Ethernet over MPLS must be in the same subnet. All other subinterfaces and backbone routers do not.
Step 5	**end** **Example:** `Router(config-subif)# end`	Exits to privileged EXEC mode.
Step 6	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 7	**source template type pseudowire** *template-name* **Example:** `Router(config-if)# source template type pseudowire atom`	Configures the source template of type pseudowire named atom
Step 8	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)#  neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 9	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.

	Command or Action	Purpose
Step 10	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 11	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 100 group GR_1 priority 2`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 12	**member pseudowire** *interface-number* **Example:** `Device(config-xconnect)# member pseudowire 1001 group GR_1 priority 2`	Specifies a second member pseudowire for redundancy.
Step 13	**member gigabitethernet** *interface-number* **Example:** `Device(config-xconnect)# member GigabitEthernet0/0/0.1 service instance 1`	Specifies the location of the Gigabit Ethernet member interface.
Step 14	**redundancy delay** *enable-delay*{*disable-delay* | **never**} **Example:** `Device(config-xconnect)# redundancy delay 0 0 group GR_1`	Specifies how long (in seconds) the backup pseudowire VC should wait to take over after the primary pseudowire VC goes down. The range is 0 to 180. Specifies how long the primary pseudowire should wait after it becomes active to take over for the backup pseudowire VC. The range is 0 to 180 seconds. If you specify the **never keyword**, the primary pseudowire VC never takes over for the backup.

Forcing a Manual Switchover to the Backup Pseudowire VC

To force the router switch over to the backup or primary pseudowire, you can enter the **xconnect backup force switchover** command in privileged EXEC mode. You can specify either the interface of the primary attachment circuit (AC) to switch to or the IP address and VC ID of the peer router.

A manual switchover can be made only if the interface or peer specified in the command is actually available and the xconnect moves to the fully active state when executing the command.

SUMMARY STEPS

1. **enable**
2. **xconnect backup force-switchover { interface** *interface-info* | **peer** *ip-address vcid}*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**xconnect backup force-switchover { interface** *interface-info* | **peer** *ip-address vcid}* **Example:** `Router# xconnect backup force-switchover peer 10.10.10.1 123`	Specifies that the router should switch to the backup or to the primary pseudowire.

Verifying the L2VPN Pseudowire Redundancy Configuration

Perform this task to verify that the L2VPN Pseudowire Redundancy feature is correctly configured.

SUMMARY STEPS

1. **show mpls l2transport vc**
2. **show xconnect all**
3. **xconnect logging redundancy**

DETAILED STEPS

Step 1 **show mpls l2transport vc**

The following is sample output from the **show mpls l2transport vc** command. In this example, the primary attachment circuit is up. The backup attachment circuit is available, but not currently selected.

Example:

```
Router# show mpls l2transport vc
Local intf     Local circuit           Dest address    VC ID      Status
-------------  ----------------------- --------------- ---------- ----------
Et0/0.1        Eth VLAN 101            10.0.0.2        101        UP
Et0/0.1        Eth VLAN 101            10.0.0.3        201        DOWN
Router# show mpls l2transport vc detail
Local interface: Et0/0.1 up, line protocol up, Eth VLAN 101 up
   Destination address 10.0.0.2 VC ID: 101, VC status UP
   .
   .
   .
Local interface: Et0/0.1 down, line protocol down, Eth VLAN 101 down
   Destination address 10.0.0.3 VC ID: 201, VC status down
   .
   .
   .
```

Step 2 **show xconnect all**

In this example, the topology is Attachment Circuit 1 to Pseudowire 1 with a Pseudowire 2 as a backup:

Example:

```
Router# show xconnect all
Legend: XC ST=Xconnect State, S1=Segment1 State, S2=Segment2 State
UP=Up, DN=Down, AD=Admin Down, IA=Inactive, NH=No Hardware
XC ST  Segment 1                         S1 Segment 2                         S2
------+---------------------------------+--+---------------------------------+--
UP pri ac   Et0/0(Ethernet)              UP mpls 10.55.55.2:1000              UP
IA sec ac   Et0/0(Ethernet)              UP mpls 10.55.55.3:1001              DN
```

In this example, the topology is Attachment Circuit 1 to Attachment Circuit 2 with a pseudowire backup for Attachment Circuit 2:

Example:

```
Router# show xconnect all
Legend: XC ST=Xconnect State, S1=Segment1 State, S2=Segment2 State
UP=Up, DN=Down, AD=Admin Down, IA=Inactive, NH=No Hardware
XC ST  Segment 1                         S1 Segment 2                         S2
------+---------------------------------+--+---------------------------------+--
UP pri ac   Se6/0:150(FR DLCI)           UP ac   Se8/0:150(FR DLCI)           UP
IA sec ac   Se6/0:150(FR DLCI)           UP mpls 10.55.55.3:7151              DN
```

Step 3 **xconnect logging redundancy**

In addition to the **show mpls l2transport vc** command and the **show xconnect** command, you can use the **xconnect logging redundancy** command to track the status of the xconnect redundancy group:

Example:

```
Router(config)# xconnect logging redundancy
```

When this command is configured, the following messages are displayed during switchover events:

Activating the primary member:

Example:

```
00:01:07: %XCONNECT-5-REDUNDANCY: Activating primary member 10.55.55.2:1000
```

Activating the backup member:

Example:

```
00:01:05: %XCONNECT-5-REDUNDANCY: Activating secondary member 10.55.55.3:1001
```

Verifying the L2VPN Pseudowire Redundancy Configuration using the commands associated with the L2VPN Protocol-Based CLIs feature

Use the following commands to verify that the L2VPN Pseudowire Redundancy feature is correctly configured.

SUMMARY STEPS

1. **show l2vpn atom vc**

2. **show l2vpn service all**
3. **logging redundancy**
4. **logging pseudowire status**

DETAILED STEPS

Step 1 **show l2vpn atom vc**

In this example, the primary attachment circuit is up. The backup attachment circuit is available, but not currently selected. The **show** output displays as follows:

Example:

```
Device# show l2vpn atom vc
Local intf     Local circuit           Dest address    VC ID      Status
-------------  ----------------------- --------------- ---------- ----------
Et0/0.1        Eth VLAN 101            10.0.0.2        101        UP
Et0/0.1        Eth VLAN 101            10.0.0.3        201        DOWN
Router# show l2vpn atom vc detail
Local interface: Et0/0.1 up, line protocol up, Eth VLAN 101 up
   Destination address 10.0.0.2 VC ID: 101, VC status UP
   .
   .
   .
Local interface: Et0/0.1 down, line protocol down, Eth VLAN 101 down
   Destination address 10.0.0.3 VC ID: 201, VC status down
   .
   .
   .
```

Step 2 **show l2vpn service all**

In this example, the topology is attachment circuit 1 to pseudowire 1 with apPseudowire 2 as a backup:

Example:

```
Device# show l2vpn service all
Legend: St=State    XC St=State in the L2VPN Service      Prio=Priority
        UP=Up       DN=Down            AD=Admin Down      IA=Inactive
        SB=Standby  HS=Hot Standby     RV=Recovering      NH=No Hardware
        m=manually selected

  Interface          Group       Encapsulation                   Prio  St  XC St
  ---------          -----       -------------                   ----  --  -----
VPWS name: foo, State: UP
  Eth1/1.1                       Eth1/1.1:100(Eth VLAN)          0     UP  UP
  pw101              blue        102.1.1.1:100(MPLS)             2     UP  UP
  pw102              blue        103.1.1.1:100(MPLS)             5     SB  IA
  pw103              blue        104.1.1.1:100(MPLS)             8     SB  IA
  pw104              blue        105.1.1.1:100(MPLS)             11    SB  IA
```

In this example, the topology is attachment circuit 1 to attachment circuit 2 with a pseudowire backup for attachment circuit 2:

Example:

```
Device# show l2vpn service all
Legend: XC ST=Xconnect State, S1=Segment1 State, S2=Segment2 State
UP=Up, DN=Down, AD=Admin Down, IA=Inactive, NH=No Hardware
```

```
XC ST  Segment 1                         S1 Segment 2                         S2
------+---------------------------------+--+----------------------------------+--
UP pri ac   Se6/0:150(FR DLCI)           UP ac   Se8/0:150(FR DLCI)           UP
IA sec ac   Se6/0:150(FR DLCI)           UP mpls 10.55.55.3:7151              DN
```

Step 3 **logging redundancy**

In addition to the **show l2vpn atom vc** command and the **show l2vpn service** command, you can use the **logging redundancy** command to enable system message log (syslog) reporting of xconnect redundancy status events:

Example:

```
Device(config)# l2vpn
Device(config-l2vpn)# logging redundancy
```

When this command is configured, the messages below will be generated during switchover events:

Activating the primary member:

Example:

```
Device(config)# l2vpn
Device(config-l2vpn)# logging pseudowire status
```

When this command is configured, this is configured the status of the pseudowire can be monitored:

Activating the primary member:

Example:

```
00:01:07: %XCONNECT-5-REDUNDANCY: Activating primary member 10.55.55.2:1000
```

Activating the backup member:

Example:

```
00:01:05: %XCONNECT-5-REDUNDANCY: Activating secondary member 10.55.55.3:1001
```

Step 4 **logging pseudowire status**

you can use the **logging pseudowire status** command to monitor the status of the pseudowire.

Example:

```
Device(config)# l2vpn
Device(config-l2vpn)# logging pseudowire status
```

Configuration Examples for L2VPN Pseudowire Redundancy

Each of the configuration examples refers to one of the following pseudowire classes:

- AToM (like-to-like) pseudowire class:

```
pseudowire-class mpls
 encapsulation mpls
```

- L2VPN IP interworking:

```
pseudowire-class mpls-ip
 encapsulation mpls
 interworking ip
```

Example L2VPN Pseudowire Redundancy and AToM (Like to Like)

The following example shows a High-Level Data Link Control (HDLC) attachment circuit xconnect with a backup pseudowire:

```
interface Serial4/0
 xconnect 10.55.55.2 4000 pw-class mpls
 backup peer 10.55.55.3 4001 pw-class mpls
```

The following example shows a Frame Relay attachment circuit xconnect with a backup pseudowire:

```
connect fr-fr-pw Serial6/0 225 l2transport
 xconnect 10.55.55.2 5225 pw-class mpls
 backup peer 10.55.55.3 5226 pw-class mpls
```

Example L2VPN Pseudowire Redundancy and L2VPN Interworking

The following example shows an Ethernet attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
interface Ethernet0/0
 xconnect 10.55.55.2 1000 pw-class mpls-ip
 backup peer 10.55.55.3 1001 pw-class mpls-ip
```

The following example shows an Ethernet VLAN attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
interface Ethernet1/0.1
 encapsulation dot1Q 200
 no ip directed-broadcast
 xconnect 10.55.55.2 5200 pw-class mpls-ip
 backup peer 10.55.55.3 5201 pw-class mpls-ip
```

The following example shows a Frame Relay attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
connect fr-ppp-pw Serial6/0 250 l2transport
 xconnect 10.55.55.2 8250 pw-class mpls-ip
 backup peer 10.55.55.3 8251 pw-class mpls-ip
```

The following example shows a PPP attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
interface Serial7/0
 encapsulation ppp
 xconnect 10.55.55.2 2175 pw-class mpls-ip
 backup peer 10.55.55.3 2176 pw-class mpls-ip
```

Example L2VPN Pseudowire Redundancy with Layer 2 Local Switching

The following example shows an Ethernet VLAN-VLAN local switching xconnect with a pseudowire backup for Ethernet segment E2/0.2. If the subinterface associated with E2/0.2 goes down, the backup pseudowire is activated:

```
connect vlan-vlan Ethernet1/0.2 Ethernet2/0.2
 backup peer 10.55.55.3 1101 pw-class mpls
```

The following example shows a Frame Relay-to-Frame Relay local switching connect with a pseudowire backup for Frame Relay segment S8/0 150. If data-link connection identifier (DLCI) 150 on S8/0 goes down, the backup pseudowire is activated:

```
connect fr-fr-ls Serial6/0 150 Serial8/0 150
 backup peer 10.55.55.3 7151 pw-class mpls
```

Example L2VPN Pseudowire Redundancy and Layer 2 Tunneling Protocol Version 3

The following example shows how to configure a backup peer for an xconnect session:

```
pseudowire-class 773
 encapsulation l2tpv3
 ip local interface GigabitEthernet0/0/0.773
!
pseudowire-class 774
 encapsulation l2tpv3
 ip local interface GigabitEthernet0/0/1.774
!
interface GigabitEthernet0/0/0.780
 encapsulation dot1Q 780
 xconnect 10.22.73.14 100 pw-class 773
  backup peer 10.22.74.14 101 pw-class 774
  backup delay 0 0
```

The following example shows how to configure a Gigabit Ethernet port with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/2
 xconnect 10.22.70.83 50 pw-class pe1-pw-primary
 backup peer 20.22.70.85 51 pw-class pe1-pw-secondary
```

The following example shows how to configure a Gigabit Ethernet VLAN with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/0.100
 encapsulation dot1q 100
 xconnect 10.22.70.83 60 pw-class pe1-pw-primary
 backup peer 10.22.70.85 61 pw-class pe1-pw-secondary
```

The following example shows how to configure a Gigabit Ethernet Q-in-Q with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/0.200
 encapsulation dot1q 200 second-dot1q 400
```

```
 xconnect 10.22.70.83 70 pw-class pe1-pw-primary
 backup peer 10.22.70.85 71 pw-class pe1-pw-secondary
```

The following example shows how to configure a Gigabit Ethernet Q-in-any with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/0.300
 encapsulation dot1q 300 second-dot1q any
 xconnect 10.22.70.83 80 pw-class pe1-pw-primary
 backup peer 10.22.70.85 81 pw-class pe1-pw-secondary
```

The following example shows how to configure an HDLC with L2VPN pseudowire redundancy and L2TPv3

```
interface Serial0/2/0:0
 no ip address
 xconnect 10.22.71.83 40 pw-class pe1-pw-hdlc
 backup peer 10.22.70.85 41 pw-class pe1-pw-hdlc-2
```

Configuration Examples for L2VPN Pseudowire Redundancy using the commands associated with the L2VPN Protocol-Based CLIs feature

Each of the configuration examples refers to one of the following interface pseudowires:

- AToM (like-to-like) interface pseudowire:

```
interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
```

- L2VPN IP interworking:

```
interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
 interworking ip
```

Example L2VPN Pseudowire Redundancy and AToM (Like to Like) using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows a High-Level Data Link Control (HDLC) attachment circuit xconnect with a backup pseudowire:

```
interface Serial4/0
 interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.55.55.3 4001
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
```

```
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
```

The following example shows a Frame Relay attachment circuit xconnect with a backup pseudowire:

```
connect fr-fr-pw Serial6/0 225 l2transport
interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.55.55.3 5226
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
```

Example L2VPN Pseudowire Redundancy and L2VPN Interworking using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows an Ethernet attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
interface Ethernet0/0
interface pseudowire 100
 source template type pseudowire ether-pw
!
l2vpn xconnect context con1
 member pseudowrire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows an Ethernet VLAN attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
interface Ethernet1/0.1
 encapsulation dot1Q 200
 no ip directed-broadcast
 interface pseudowire 100
 source template type pseudowire ether-pw
!
l2vpn xconnect context con1
 member pseudowrire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows a Frame Relay attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
connect fr-ppp-pw Serial6/0 250 l2transport
 interface pseudowire 100
 source template type pseudowire ether-pw
!
l2vpn xconnect context con1
 member pseudowrire 100 group GR_1 priority 1
```

```
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows a PPP attachment circuit xconnect with L2VPN IP interworking and a backup pseudowire:

```
interface Serial7/0
 encapsulation ppp
 interface pseudowire 100
 source template type pseudowire ether-pw
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

Example L2VPN Pseudowire Redundancy and Layer 2 Tunneling Protocol Version 3 using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows how to configure a backup peer for an xconnect session:

```
interface pseudowire 773
 encapsulation l2tpv3
 ip local interface GigabitEthernet0/0/0.773
!
interface pseudowire 774
 encapsulation l2tpv3
 ip local interface GigabitEthernet0/0/1.774
!
interface GigabitEthernet0/0/0.780
 encapsulation dot1Q 780
 interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.22.73.14 100
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows how to configure a Gigabit Ethernet port with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/2
 interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.22.70.83 50
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
```

```
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows how to configure a Gigabit Ethernet VLAN with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/0.100
 encapsulation dot1q 100
 interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.22.70.83 60
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows how to configure a Gigabit Ethernet Q-in-Q with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/0.200
 encapsulation dot1q 200 second-dot1q 400
 interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.22.70.83 70
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows how to configure a Gigabit Ethernet Q-in-any with L2VPN pseudowire redundancy and L2TPv3:

```
interface GigabitEthernet0/0/0.300
 encapsulation dot1q 300 second-dot1q any
 interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.22.70.83 80
!
l2vpn xconnect context con1
 member pseudowire 100 group GR_1 priority 1
 member pseudowire 1001 group GR_1 priority 2
 member GigabitEthernet0/0/2 service-instance 1
 redundancy delay 0 0 group GR_1
 interworking ip
```

The following example shows how to configure an HDLC with L2VPN pseudowire redundancy and L2TPv3

```
interface Serial0/2/0:0
 no ip address
 interface pseudowire 100
 source template type pseudowire ether-pw
 neighbor 10.22.71.83 40
!
l2vpn xconnect context con1
```

```
l2vpn xconnect context con1
member pseudowire 100 group GR_1 priority 1
member pseudowire 1001 group GR_1 priority 2
member GigabitEthernet0/0/2 service-instance 1
redundancy delay 0 0 group GR_1
interworking ip
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Wide-area networking commands	*Cisco IOS Wide-Area Networking Command Reference*
Cisco IOS XE Multiprotocol Label Switching configuration tasks	*Cisco IOS XE Multiprotocol Label Switching Configuration Guide*
Cisco IOS XE Wide-area networking configuration tasks	*Cisco IOS XE Wide-Area Networking Configuration Guide*

Standards

Standards	Title
None	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
None	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for L2VPN Pseudowire Redundancy

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 25: Feature Information for L2VPN Pseudowire Redundancy

Feature Name	Releases	Feature Information
L2VPN Pseudowire Redundancy	XE 2.3 XE 3.3S	This feature enables you to set up your network to detect a failure in the network and reroute the Layer 2 service to another endpoint that can continue to provide service. In Cisco IOS XE Release 2.3, this feature was integrated into the Cisco ASR 1000 Series Aggregation Service Routers. In Cisco IOS XE Release 3.3S, this feature supports Layer 2 Tunneling Protocol Version 3 (L2TPv3). The following commands were introduced or modified: **backup delay (L2VPN local switching), backup peer, show xconnect, xconnect backup force-switchover, xconnect logging redundancy.**
L2VPN Pseudowire Redundancies	Cisco IOS XE Fuji 16.9.1	In Cisco IOS XE Fuji 16.9.1, this feature is supported on Cisco 1000 Series ISRs.

CHAPTER 14

Pseudowire Group Switchover

The Pseudowire Group Switchover feature allows all pseudowires in a group to be quickly switched over to backup pseudowires. This group switchover is triggered by a single “group down” status message received from a remote peer.

CHAPTER 15

L2VPN Pseudowire Switching

This feature module explains how to configure L2VPN Pseudowire Switching, which extends layer 2 virtual private network (L2VPN) pseudowires across an interautonomous system (inter-AS) boundary or across two separate multiprotocol label switching (MPLS) networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for L2VPN Pseudowire Switching

- In Cisco IOS XE Release 2.4, Pseudowire Switching is supported on Ethernet over MPLS attachment circuits.
- L2VPN Pseudowire Switching is supported with AToM.
- Only static, on-box provisioning is supported.
- Sequencing numbers in AToM packets are not processed by L2VPN Pseudowire Switching. The feature blindly passes the sequencing data through the xconnect packet paths, a process that is called transparent sequencing. The endpoint PE-CE connections enforce the sequencing.

- You can ping the adjacent next-hop PE router. End-to-end LSP pings are not supported.
- Do not configure IP or Ethernet interworking on a router where L2VPN Pseudowire Switching is enabled. Instead, configure interworking on the routers at the edge PEs of the network.
- The control word negotiation results must match. If either segment does not negotiate the control word, the control word is disabled for both segments.
- AToM Graceful Restart is negotiated independently on each pseudowire segment. If there is a transient loss of the LDP session between two AToM PE routers, packets continue to flow.
- Per-pseudowire quality of service (QoS) is not supported. Traffic Engineering (TE) tunnel selection is supported.
- Attachment circuit interworking is not supported.

Information About L2VPN Pseudowire Switching

How L2VPN Pseudowire Switching Works

L2VPN Pseudowire Switching allows the user to extend L2VPN pseudowires across an inter-AS boundary or across two separate MPLS networks, as shown in the figures below. L2VPN Pseudowire Switching connects two or more contiguous pseudowire segments to form an end-to-end multihop pseudowire. This end-to-end pseudowire functions as a single point-to-point pseudowire.

As shown in the second figure below, L2VPN Pseudowire Switching enables you to keep the IP addresses of the edge PE routers private across inter-AS boundaries. You can use the IP address of the autonomous system boundary routers (ASBRs) and treat them as pseudowire aggregation (PE-agg) routers. The ASBRs join the pseudowires of the two domains.

L2VPN Pseudowire Switching also enables you to keep different administrative or provisioning domains to manage the end-to-end service. At the boundaries of these networks, PE-agg routers delineate the management responsibilities.

Figure 33: L2VPN Pseudowire Switching in an Intra-AS Topology

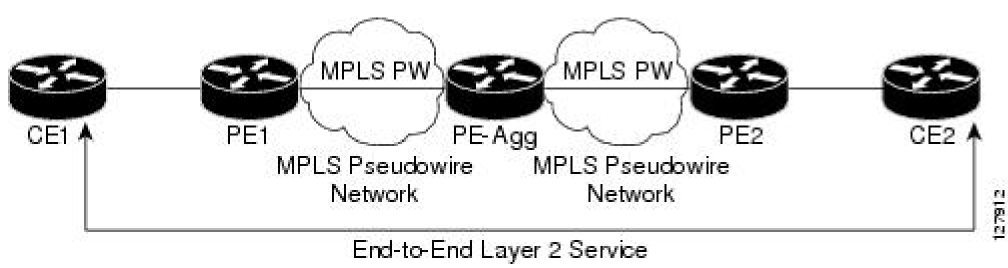
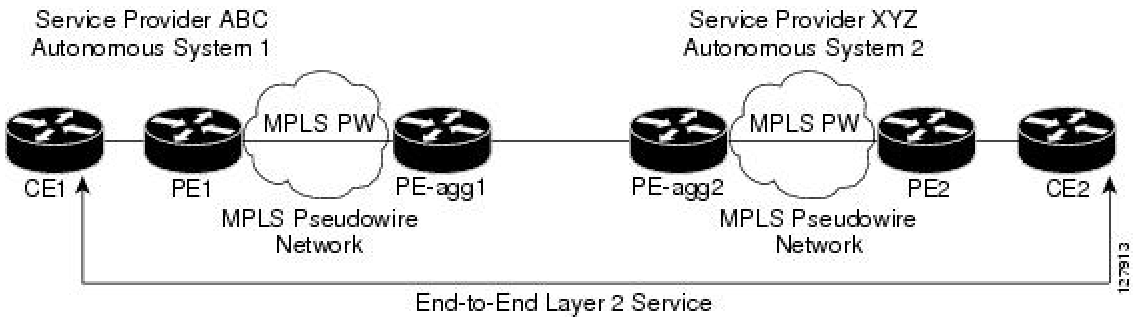


Figure 34: L2VPN Pseudowire Switching in an Inter-AS Topology



How Packets Are Manipulated at the Aggregation Point

Switching AToM packets between two AToM pseudowires is the same as switching any MPLS packet. The MPLS switching data path switches AToM packets between two AToM pseudowires. The following list explains exceptions:

- The outgoing virtual circuit (VC) label replaces the incoming VC label in the packet. New Internal Gateway Protocol (IGP) labels and Layer 2 encapsulation are added.
- The incoming VC label time-to-live (TTL) field is decremented by one and copied to the outgoing VC label TTL field.
- The incoming VC label EXP value is copied to the outgoing VC label EXP field.
- The outgoing VC label 'Bottom of Stack' S bit in the outgoing VC label is set to1.
- AToM control word processing is not performed at the L2VPN Pseudowire Switching aggregation point. Sequence numbers are not validated. Use the Router Alert label for LSP Ping; do not require control word inspection to determine an LSP Ping packet.

How to Configure L2VPN Pseudowire Switching

Configuring

Use the following procedure to configure L2VPN Pseudowire Switching on each of the PE-agg routers.

Before you begin

- This procedure assumes that you have configured basic AToM L2VPNs. This procedure does not explain how to configure basic AToM L2VPNs that transport Layer 2 packets over an MPLS backbone. For information on the basic configuration, see Any Transport over MPLS.
- For inter-Autonomous configurations, ASBRs require a labeled interface.

Note In this configuration, you are limited to two **neighbor**commands after entering the **l2 vfi**command.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *name* **point-to-point**
4. **neighbor** *ip-address* *vcid* **encapsulation mpls** | **pw-class** *pw-class-name*
5. **exit**
6. **exit**
7. **show mpls l2transport vc** [**vcid** [*vc-id* | [*vc-id-min vc-id-max*]] [**interface** *name*[*local-circuit-id*]] [**destination** *ip-address* | *name*] [**detail**]
8. **show vfi** [*vfi-name*]
9. **ping** [*protocol*] [**tag**] {*host-name*| *system-address*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *name* **point-to-point** **Example:** `Router(config)# l2 vfi atomtunnel point-to-point`	Creates a point-to-point Layer 2 virtual forwarding interface (VFI) and enters VFI configuration mode.
Step 4	**neighbor** *ip-address* *vcid* **encapsulation mpls** | **pw-class** *pw-class-name* **Example:** `Router(config-vfi)# neighbor 10.0.0.1 100 pw-class mpls`	Sets up an emulated VC. Specify the IP address and the VC ID of the remote router. Also specify the pseudowire class to use for the emulated VC. **Note** Only two **neighbor**commands are allowed for each **l2 vfi point-to-point** command.
Step 5	**exit** **Example:** `Router(config-vfi)# exit`	Exits VFI configuration mode.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 7	**show mpls l2transport vc** [**vcid** [*vc-id* | [*vc-id-min* *vc-id-max*]] [**interface** *name*[*local-circuit-id*]] [**destination** *ip-address* | *name*] [**detail**] **Example:** `Router# show mpls l2transport vc`	Verifies that the L2VPN Pseudowire Switching session has been established.
Step 8	**show vfi** [*vfi-name*] **Example:** `Router# show vfi atomtunnel`	Verifies that a point-to-point VFI has been established.
Step 9	**ping** [*protocol*] [**tag**] {*host-name*| *system-address*} **Example:** `Router# ping 10.1.1.1`	When issued from the CE routers, this command verifies end-to-end connectivity.

Examples

The following example displays the output of the **show mpls l2transport vc** command:

```
Router# show mpls l2transport vc
Local intf     Local circuit              Dest address    VC ID Status
-------------  -------------------------- --------------- ----- ----
MPLS PW        10.0.1.1:100               10.0.1.1        100   UP
MPLS PW        10.0.1.1:100               10.0.1.1        100   UP
```

The following example displays the output of the **show vfi**command:

```
Router# show vfi
VFI name: test, type: point-to-point
 Neighbors connected via pseudowires:
   Router ID        Pseudowire ID
   10.0.1.1         100
   10.0.1.1         100
```

How to Configure L2VPN Pseudowire Switching using the commands associated with the L2VPN Protocol-Based CLIs feature

Perform this task to configure L2VPN Pseudowire Switching on each of the PE-agg routers. In this configuration, you are limited to two **neighbor** commands after entering the **l2vpn xconnect** command.

Before you begin

- This task assumes that you have configured basic AToM L2VPNs. This task does not explain how to configure basic AToM L2VPNs that transport Layer 2 packets over an MPLS backbone. For information on the basic configuration, see the “Any Transport over MPLS” section.
- For interautonomous configurations, autonomous system boundary routers (ASBRs) require a labeled interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *number*
4. **encapsulation mpls**
5. **neighbor** *peer-address vcid-value*
6. **exit**
7. **interface pseudowire** *number*
8. **encapsulation mpls**
9. **neighbor** *peer-address vcid-value*
10. **exit**
11. **l2vpn xconnect context** *context-name*
12. **member pseudowire** *interface-number*
13. **member** *ip-address vcid* **encapsulation mpls**
14. **member pseudowire** *interface-number*
15. **member** *ip-address vcid* **encapsulation mpls**
16. **exit**
17. **exit**
18. **show l2vpn atom vc** [**vcid** [*vc-id* | *vc-id-min vc-id-max*]] [**interface** *type number* [*local-circuit-id*]] [**destination** *ip-address* | *name*] [**detail**]
19. **ping** [*protocol*] [**tag**] {*hostname*| *system-address*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 4	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 5	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 7	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 200`	Specifies the pseudowire interface and enters interface configuration mode.
Step 8	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 9	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.0.0.2 124`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 11	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 12	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 13	**member** *ip-address vcid* **encapsulation mpls** **Example:** `Device(config-xconnect)# member 10.0.0.1 123 encapsulation mpls`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection. **Note** Only two **member** commands are allowed for each **l2vpn xconnect context** command.
Step 14	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 200`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 15	**member** *ip-address vcid* **encapsulation mpls** **Example:** `Device(config-xconnect)# member 10.0.0.2 124 encapsulation mpls`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection. **Note** Only two **member** commands are allowed for each **l2vpn xconnect context** command.
Step 16	**exit** **Example:** `Device(config-xconnect)# exit`	Exits Xconnect configuration mode.
Step 17	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.
Step 18	**show l2vpn atom vc** [**vcid** [*vc-id* | *vc-id-min vc-id-max*]] [**interface** *type number* [*local-circuit-id*]] [**destination** *ip-address* | *name*] [**detail**] **Example:** `Device# show l2vpn atom vc`	Displays information about Any Transport over MPLS (AToM) virtual circuits (VCs) and static pseudowires that have been enabled to route Layer 2 packets on a device.
Step 19	**ping** [*protocol*] [**tag**] {*hostname*| *system-address*} **Example:**	When issued from the CE routers, verifies end-to-end connectivity.

	Command or Action	Purpose
	`Device# ping 10.1.1.1`	

Configuring

Use the following procedure to configure L2VPN Pseudowire Switching on each of the PE-agg routers.

Before you begin

- This procedure assumes that you have configured basic AToM L2VPNs. This procedure does not explain how to configure basic AToM L2VPNs that transport Layer 2 packets over an MPLS backbone. For information on the basic configuration, see Any Transport over MPLS.
- For inter-Autonomous configurations, ASBRs require a labeled interface.

Note In this configuration, you are limited to two **neighbor**commands after entering the **l2 vfi**command.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *name* **point-to-point**
4. **neighbor** *ip-address vcid* **encapsulation mpls** | **pw-class** *pw-class-name*
5. **exit**
6. **exit**
7. **show mpls l2transport vc** [**vcid** [*vc-id* | [*vc-id-min vc-id-max*]] [**interface** *name*[*local-circuit-id*]] [**destination** *ip-address* | *name*] [**detail**]
8. **show vfi** [*vfi-name*]
9. **ping** [*protocol*] [**tag**] {*host-name*| *system-address*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**l2 vfi** *name* **point-to-point** **Example:** `Router(config)# l2 vfi atomtunnel point-to-point`	Creates a point-to-point Layer 2 virtual forwarding interface (VFI) and enters VFI configuration mode.
Step 4	**neighbor** *ip-address* *vcid* **encapsulation mpls** | **pw-class** *pw-class-name* **Example:** `Router(config-vfi)# neighbor 10.0.0.1 100 pw-class mpls`	Sets up an emulated VC. Specify the IP address and the VC ID of the remote router. Also specify the pseudowire class to use for the emulated VC. **Note** Only two **neighbor**commands are allowed for each **l2 vfi point-to-point** command.
Step 5	**exit** **Example:** `Router(config-vfi)# exit`	Exits VFI configuration mode.
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 7	**show mpls l2transport vc** [**vcid** [*vc-id* | [*vc-id-min* *vc-id-max*]] [**interface** *name*[*local-circuit-id*]] [**destination** *ip-address* | *name*] [**detail**] **Example:** `Router# show mpls l2transport vc`	Verifies that the L2VPN Pseudowire Switching session has been established.
Step 8	**show vfi** [*vfi-name*] **Example:** `Router# show vfi atomtunnel`	Verifies that a point-to-point VFI has been established.
Step 9	**ping** [*protocol*] [**tag**] {*host-name*| *system-address*} **Example:** `Router# ping 10.1.1.1`	When issued from the CE routers, this command verifies end-to-end connectivity.

Examples

The following example displays the output of the **show mpls l2transport vc** command:

```
Router# show mpls l2transport vc
Local intf     Local circuit              Dest address    VC ID Status
-------------  -------------------------- --------------- ----- ----
```

```
MPLS PW       10.0.1.1:100               10.0.1.1       100   UP
MPLS PW       10.0.1.1:100               10.0.1.1       100   UP
```

The following example displays the output of the **show vfi**command:

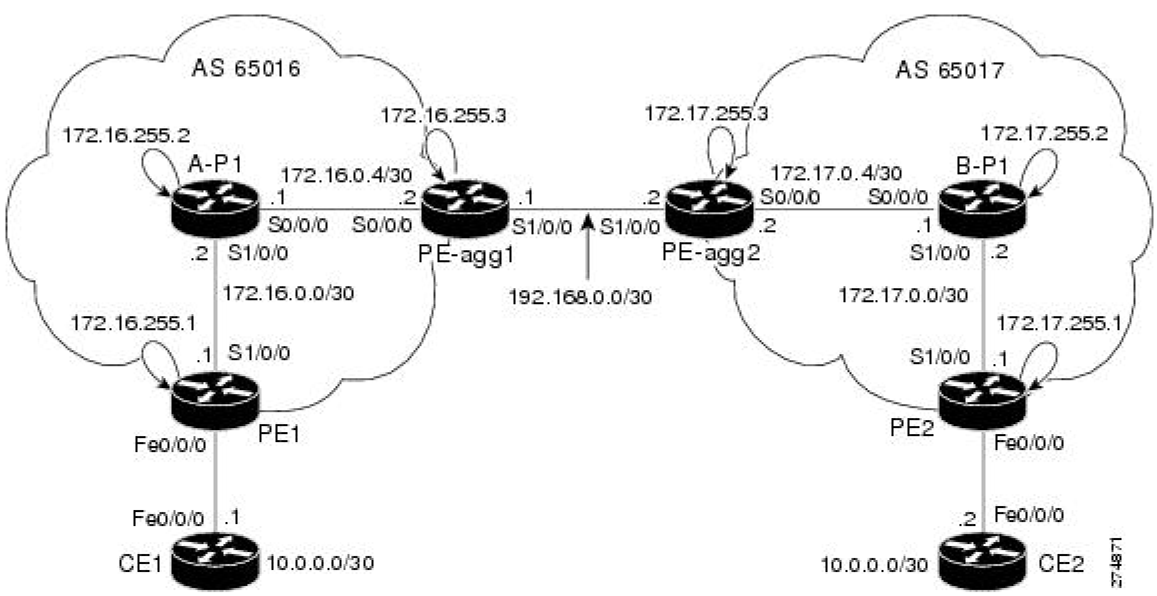
```
Router# show vfi
VFI name: test, type: point-to-point
 Neighbors connected via pseudowires:
   Router ID        Pseudowire ID
   10.0.1.1         100
   10.0.1.1         100
```

Configuration Examples for L2VPN Pseudowire Switching

L2VPN Pseudowire Switching in an Inter-AS Configuration Example

Two separate autonomous systems are able to pass L2VPN packets, because the two PE-agg routers have been configured with L2VPN Pseudowire Switching. This example configuration is shown in the figure below.

Figure 35: L2VPN Pseudowire Switching in an InterAutonomous System



CE1	CE2

CE1	CE2
version 12.0 service timestamps debug uptime service timestamps log uptime service password-encryption ! hostname [ce1] ! boot-start-marker boot-end-marker ! enable secret 5 1o9N6$LSrxHufTn0vjCY0nW8hQX. ! ip subnet-zero ip cef no ip domain-lookup ! interface FastEthernet0/0/0 ip address 10.0.0.1 255.255.255.252 no ip directed-broadcast ! ip classless ! control-plane !	version 12.0 service timestamps debug uptime service timestamps log uptime service password-encryption ! hostname [ce2] ! boot-start-marker boot-end-marker ! enable secret 5 1YHo6$LQ4z5PdrF5B9dnL75Xvvm1 ! ip subnet-zero ip cef no ip domain-lookup ! interface FastEthernet0/0/0 ip address 10.0.0.2 255.255.255.252 no ip directed-broadcast ! ip classless ! control-plane !

CE1	CE2

`line con 0`
 	
` exec-timeout 0 0`
 	
`line aux 0`
 	
`line vty 0 4`
 	
` login`
 	
`!`
 	
`no cns aaa enable`
 	
`end`
	`line con 0`
	
	` exec-timeout 0 0`
	
	`line aux 0`
	
	`line vty 0 4`
	
	` login`
	
	`!`
	
	`no cns aaa enable`
	
	`end`

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference
L2VPN pseudowire redundancy	"L2VPN Pseudowire Redundancy" feature module in the *MPLS Layer 2 VPNs Configuration Guide*.
H-VPLS	"Configuring VPLS" in the "Configuring Multiprotocol Label Switching on the Optical Services Modules" chapter in the *Optical Services Modules Installation and Configuration Notes*, 12.2SR document.
MPLS traffic engineering	"MPLS Traffic Engineering Fast Reroute Link and Node Protection" feature module in the *MPLS Traffic Engineering: Path, Link, and Node Protection Configuration Guide* (part of the Multiprotocol Label Switching Configuration Guide Library)

Standards

Standard	Title
http://www.ietf.org/rfc/rfc4447.txt	*Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)*

Standard	Title
http://www3.ietf.org/proceedings/06mar/IDs/draft-ietf-l2vpn-vpls-ldp-08.txt	*Virtual Private LAN Services over MPLS*
http://www.ietf.org/internet-drafts/draft-ietf-pwe3-segmented-pw-02.txt	*Segmented Pseudo Wire*
draft-ietf-pwe3-vccv-10.txt	*Pseudo Wire Virtual Circuit Connectivity Verification (VCCV)*
draft-ietf-pwe3-oam-msg-map-03.txt	*Pseudo Wire (PW) OAM Message Mapping*

MIBs

MIB	MIBs Link
Pseudowire Emulation Edge-to-Edge MIBs for Ethernet, Frame Relay, and ATM Services	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for L2VPN Pseudowire Switching

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 26: Feature Information for L2VPN Pseudowire Switching

Feature Name	Releases	Feature Information
L2VPN Pseudowire Switching	Cisco IOS XE Release 2.4	The L2VPN Pseudowire Switching feature extends layer 2 virtual private network (L2VPN) pseudowires across an interautonomous system (inter-AS) boundary or across two separate multiprotocol label switching (MPLS) networks. In Cisco IOS XE Release 2.4, the L2VPN Pseudowire Switching feature is supported with Ethernet over MPLS. The following commands were introduced or modified: **l2 vfi point-to-point**, **neighbor**(L2VPN Pseudowire Switching), **show vfi**.
L2VPN Pseudowire-Switching	Cisco IOS XE Fuji 16.9.1	In Cisco IOS XE Fuji 16.9.1, the L2VPN Pseudowire Switching feature is supported on Cisco 1000 Series ISRs.

CHAPTER 16

Xconnect as a Client of BFD

The Xconnect as a Client of Bidirectional Forwarding Detection (BFD) feature provides a trigger for redundant pseudowire switchover based on BFD's fast failure detection capabilities.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Xconnect as a Client of BFD

Xconnect as a Client of BFD

Redundant pseudowires are deployed to provide fault tolerance and resiliency to L2VPN-backhauled connections. The speed at which a system recovers from failures, especially when scaled to large numbers of pseudowires, is critical to many service providers and service level agreements (SLAs). The configuration of a trigger for redundant pseudowire switchover reduces the time that it takes a large number of pseudowires to failover. A fundamental component of bidirectional forwarding detection (BFD) capability is enabled by fast-failure detection (FFD).

The configuration of this feature refers to a BFD configuration, such as the following (the second URL in the **bfd map** command is the loopback URL in the **monitor peer bfd** command):

```
bfd-template multi-hop mh
 interval min-tx 200 min-rx 200 multiplier 3 !
bfd map ipv4  10.1.1.0/24  10.1.1.1/32 mh
```

How to Configure Xconnect as a Client of BFD

Configuring Xconnect as a Client of BFD

Perform this task to configure a trigger for redundant pseudowire switchover.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class mpls-ffd**
   - Enters pseudowire class configuration mode.
4. **encapsulation mpls**
5. **monitor peer bfd** [**local interface** *interface-type interface-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pseudowire-class mpls-ffd** • Enters pseudowire class configuration mode. **Example:** `Device(config)# pseudowire-class mpls-ffd`	Establishes a pseudowire class for MPLS fast-failure detection.
Step 4	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation to be MPLS.
Step 5	**monitor peer bfd** [**local interface** *interface-type interface-number*]	Enables the pseudowire fast-failure detection capability.

	Command or Action	Purpose
	Example: `Device(config-pw-class)# monitor peer bfd local interface loopback 0`	

Configuration Examples for Xconnect as a Client of BFD

Example: Xconnect as a Client of BFD

Pseudowire Class Configuration

The following example shows pseudowire fast-failure detection enabled for a pseudowire class:

```
pseudowire-class mpls-ffd
 encapsulation mpls
 monitor peer bfd local interface Loopback0
```

Template Configuration

The following example shows pseudowire fast-failure detection enabled in a template:

```
template type pseudowire 1
 encapsulation mpls
 monitor peer bfd local interface Ethernet0/1
```

Interface Configuration

The following example shows pseudowire fast-failure detection enabled for an interface:

```
interface pseudowire100
 encapsulation mpls
 neighbor 10.10.1.1 21190
 monitor peer bfd local interface Ethernet0/1
```

Additional References

Related Documents

Related Topic	Document Title
Any Transport over MPLS	Any Transport over MPLS
High Availability for AToM	AToM Graceful Restart
L2VPN Interworking	L2VPN Interworking

Related Topic	Document Title
Layer 2 local switching	Layer 2 Local Switching
PWE3 MIB	Pseudowire Emulation Edge-to-Edge MIBs for Ethernet and Frame Relay Services
Packet sequencing	Any Transport over MPLS (AToM) Sequencing Support
BFD configuration	IP Routing BFD Configuration Guide

Standards

Standards	Title
None	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
None	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Xconnect as a Client of BFD

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 27: Feature Information for Xconnect as a Client of BFD

Feature Name	Releases	Feature Information
Xconnect as a Client of BFD	Cisco IOS XE Release 3.8S	This feature provides fast-failure detection for L2VPN pseudowire redundancy. The following command was introduced: **monitor peer bfd**.

CHAPTER 17

H-VPLS N-PE Redundancy for QinQ Access

The H-VPLS N-PE Redundancy for QinQ Access feature enables two network provider edge (N-PE) devices to provide failover services to a user provider edge (U-PE) device in a hierarchical virtual private LAN service (H-VPLS). Having redundant N-PE devices provides improved stability and reliability against link and node failures.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for H-VPLS N-PE Redundancy for QinQ Access

- Before configuring this feature, configure your hierarchical virtual private LAN service (H-VPLS) network and make sure it is operating correctly.
- Make sure that the PE-to-customer edge (CE) interface is configured with a list of allowed VLANs.
- To provide faster convergence, you can enable the MPLS Traffic Engineering—Fast Reroute feature in the Multiprotocol Label Switching (MPLS) core.

- Enable the L2VPN Pseudowire Redundancy feature on the user provider edge (U-PE) devices for MPLS access.
- When configuring Multiple Spanning Tree Protocol (MSTP), specify that one of the network provider edge (N-PE) devices is the root by assigning it the lowest priority using the **spanning-tree mst** *instance-id* **priority** *priority* command.
- When configuring MSTP, make sure that each device participating in the spanning tree is in the same region and is the same revision by issuing the **revision**, **name**, and **instance** commands in MST configuration mode.

Restrictions for H-VPLS N-PE Redundancy for QinQ Access

- This feature cannot be used with the VPLS Autodiscovery feature on pseudowires that attach to network provider edge (N-PE) devices. When you create the virtual private LAN service (VPLS), you can manually create the virtual forwarding instance (VFI).
- You cannot configure more than one pseudowire to carry the bridge protocol data unit (BPDU) packets between two redundant network provider edge (N-PE) devices on the same Virtual Private LAN service (VPLS) site.
- You cannot configure a local loopback address as a neighbor when you configure the H-VPLS N-PE Redundancy feature on N-PE devices. If you do so, the following error message is displayed:

```
VPLS local switching to peer address not supported
```

- Only two N-PE devices can be connected to each U-PE device.
- The spanning-tree mode must be Multiple Spanning Tree Protocol (MSTP) for the H-VPLS N-PE Redundancy feature. If the spanning-tree mode changes, the H-VPLS N-PE Redundancy feature might not work correctly, even though the pseudowire that carries the BPDU packet still exists and the H-VPLS N-PE Redundancy feature is still configured.

Information About H-VPLS N-PE Redundancy for QinQ Access

How H-VPLS N-PE Redundancy for QinQ Access Works

In a network configured with the H-VPLS N-PE Redundancy feature, the user provider edge (U-PE) device is connected to two network provider edge (N-PE) devices. This feature provides a level of redundancy that can tolerate both link and device faults. If a failure occurs in the network that disables one N-PE device from transmitting data, the other N-PE device takes over. This feature works with both QinQ access based on Multiple Spanning Tree Protocol (MSTP) and Multiprotocol Label Switching (MPLS) access based on pseudowire redundancy.

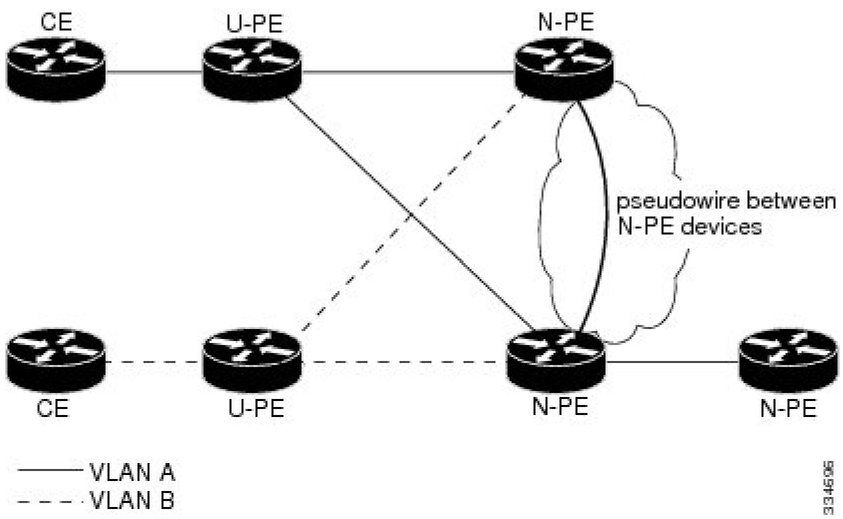
H-VPLS N-PE Redundancy with QinQ Access Based on MSTP

The H-VPLS N-PE Redundancy with QinQ Access feature uses the Multiple Spanning Tree Protocol (MSTP) running on the network provider edge (N-PE) devices and user provider edge (U-PE) devices in a hierarchical

Virtual Private LAN service (H-VPLS) network. A pseudowire running between N-PE devices carries only MSTP bridge protocol data units (BPDUs). The pseudowire running between the N-PE devices is always up and is used to create a loop path between N-PE devices so that MSTP blocks one of the redundant paths between the U-PE device and the N-PE devices. If the primary N-PE device or the path to it fails, MSTP enables the path to the backup N-PE device.

The figure below shows an H-VPLS network with redundant access. Each U-PE device has two connections, one to each N-PE device. Between the two N-PE devices is a pseudowire to provide a loop path for MSTP BPDUs. The network topology allows for the backup N-PE device to take over if the primary N-PE device or the path to it fails.

Figure 36: H-VPLS N-PE Redundancy with QinQ Access Based on MSTP



How to Configure H-VPLS N-PE Redundancy for QinQ Access

Configuring the VPLS Pseudowire Between the N-PE Devices using the commands associated with the L2VPN Protocol-Based CLIs feature

Configuring network provider edge (N-PE) redundancy in a hierarchical Virtual Private LAN service (H-VPLS) network requires that you configure the VPLS pseudowire for transporting bridge protocol data unit (BPDU) packets. For the core pseudowire between the N-PE devices, you configure a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) and attach the VFI to a bridge-domain (described here). Then, in the next task, you bind the service instance to the bridge-domain. This configuration provides a redundancy that provides improved reliability against link and node failures.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *name*
4. **vpn id** *vpn id*
5. **member** *ip-address* **encapsulation mpls**

6. **forward permit l2protocol all**
7. **exit**
8. **bridge-domain** *bridge-id*
9. **member vfi** *vfi-name*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *name* **Example:** `Device(config)# l2vpn vfi context VPLS-10`	Establishes a L2VPN VFI between two or more separate networks, and enters L2VFI configuration mode.
Step 4	**vpn id** *vpn id* **Example:** `Device(config-vfi)# vpn id 10`	Sets a VPN ID on the Virtual Private LAN Services (VPLS) instance. • Use the same VPN ID for the PE devices that belong to the same VPN. • Make sure the VPN ID is unique for each VPN in the service provider network. The range is from 1 to 4294967295.
Step 5	**member** *ip-address* **encapsulation mpls** **Example:** `Device(config-vfi)# member 102.102.102.102 encapsulation mpls`	Specifies the devices that form a point-to-point L2VPN VFI connection. • *ip-address*—IP address of the VFI neighbor. • **encapsulation mpls**—Specifies Multiprotocol Label Switching (MPLS) as the data encapsulation method.
Step 6	**forward permit l2protocol all** **Example:** `Device(config-vfi)# forward permit l2protocol all`	Creates a pseudowire that is to be used to transport BPDU packets between the two N-PE devices.
Step 7	**exit** **Example:** `Device(config-vfi)# exit`	Returns to global configuration mode.
Step 8	**bridge-domain** *bridge-id* **Example:**	Configures components on a bridge domain, and enters bridge-domain configuration mode.

	Command or Action	Purpose
	`Device(config)# bridge-domain 10`	
Step 9	**member vfi** *vfi-name* **Example:** `Device(config-bdomain)# member vfi VPLS-10`	Configures the VFI member in the bridge-domain.
Step 10	**end** **Example:** `Device(config-bdomain)# end`	Returns to privileged EXEC mode.

Configuring the VPLS Pseudowire Between the N-PE Devices using the commands associated with the L2VPN Protocol-Based CLIs feature

Configuring network provider edge (N-PE) redundancy in a hierarchical Virtual Private LAN service (H-VPLS) network requires that you configure the VPLS pseudowire for transporting bridge protocol data unit (BPDU) packets. For the core pseudowire between the N-PE devices, you configure a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) and attach the VFI to a bridge-domain (described here). Then, in the next task, you bind the service instance to the bridge-domain. This configuration provides a redundancy that provides improved reliability against link and node failures.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *name*
4. **vpn id** *vpn id*
5. **member** *ip-address* **encapsulation mpls**
6. **forward permit l2protocol all**
7. **exit**
8. **bridge-domain** *bridge-id*
9. **member vfi** *vfi-name*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**l2vpn vfi context** *name* **Example:** `Device(config)# l2vpn vfi context VPLS-10`	Establishes a L2VPN VFI between two or more separate networks, and enters L2VFI configuration mode.
Step 4	**vpn id** *vpn id* **Example:** `Device(config-vfi)# vpn id 10`	Sets a VPN ID on the Virtual Private LAN Services (VPLS) instance. • Use the same VPN ID for the PE devices that belong to the same VPN. • Make sure the VPN ID is unique for each VPN in the service provider network. The range is from 1 to 4294967295.
Step 5	**member** *ip-address* **encapsulation mpls** **Example:** `Device(config-vfi)# member 102.102.102.102 encapsulation mpls`	Specifies the devices that form a point-to-point L2VPN VFI connection. • *ip-address*—IP address of the VFI neighbor. • **encapsulation mpls**—Specifies Multiprotocol Label Switching (MPLS) as the data encapsulation method.
Step 6	**forward permit l2protocol all** **Example:** `Device(config-vfi)# forward permit l2protocol all`	Creates a pseudowire that is to be used to transport BPDU packets between the two N-PE devices.
Step 7	**exit** **Example:** `Device(config-vfi)# exit`	Returns to global configuration mode.
Step 8	**bridge-domain** *bridge-id* **Example:** `Device(config)# bridge-domain 10`	Configures components on a bridge domain, and enters bridge-domain configuration mode.
Step 9	**member vfi** *vfi-name* **Example:** `Device(config-bdomain)# member vfi VPLS-10`	Configures the VFI member in the bridge-domain.
Step 10	**end** **Example:** `Device(config-bdomain)# end`	Returns to privileged EXEC mode.

Binding the Service Instance to the Bridge-Domain

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **exit**
7. **bridge-domain** *bridge-id*
8. **member** *interface-type-number* **service-instance** *service-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet0/1/0`	Specifies the interface to configure, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Configures an Ethernet service instance on the interface, and enters Ethernet service configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 10`	Enables IEEE 802.1Q encapsulation of traffic on the specified interface in a VLAN.
Step 6	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to global configuration mode.
Step 7	**bridge-domain** *bridge-id* **Example:** `Device(config)# bridge-domain 10`	Configures components on the bridge domain, and enters bridge-domain configuration mode.

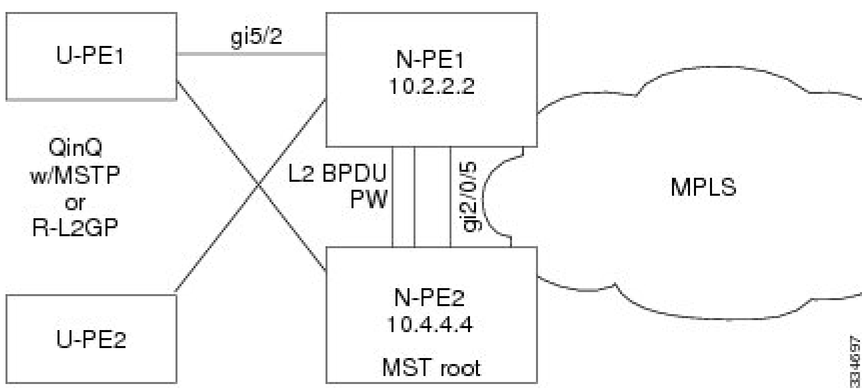
	Command or Action	Purpose
Step 8	**member** *interface-type-number* **service-instance** *service-id* **Example:** `Device(config-bdomain)# member GigabitEthernet0/1/0 service-instance 10`	Binds the service instance to the bridge-domain instance.
Step 9	**end** **Example:** `Device(config-bdomain)# end`	Returns to privileged EXEC mode.

Configuration Examples for H-VPLS N-PE Redundancy for QinQ Access

Example: H-VPLS N-PE Redundancy for QinQ Access

The figure below shows a configuration that is set up for the H-VPLS N-PE Redundancy with QinQ Access feature.

Figure 37: H-VPLS N-PE Redundancy with QinQ Access Topology



The table below shows the configuration of two network provider edge (N-PE) devices.

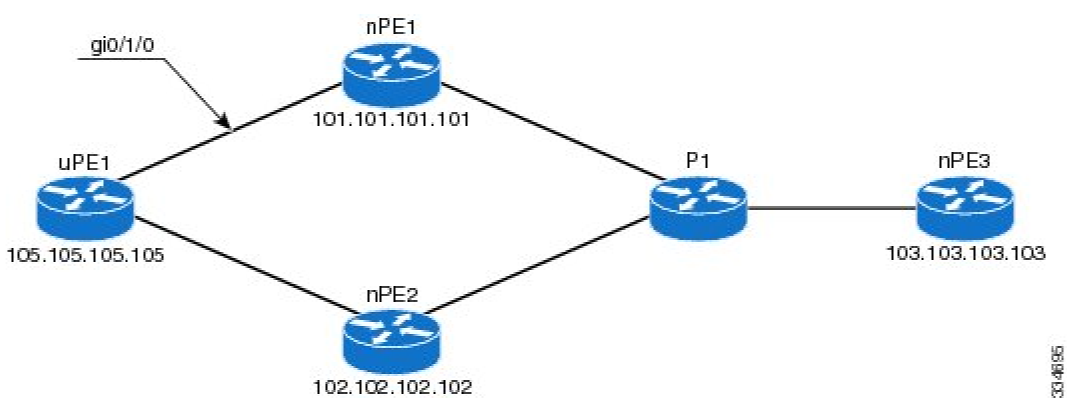
Table 28: Example: H-VPLS N-PE Redundancy for QinQ Access

N-PE1	N-PE2
l2vpn vfi context VPLS-10 vpn id 10 member 10.4.4.4 encapsulation mpls forward permit l2protocol all ! bridge-domain 10 member vfi VPLS-10 member GigabitEthernet5/2 service-instance 10 ! interface GigabitEthernet5/2 service instance 10 ethernet encapsulation dot1q 10 ! spanning-tree mode mst spanning-tree extend system-id ! spanning-tree mst configuration name myMstName revision 10 instance 1 vlan 10	l2vpn vfi context VPLS-10 vpn id 10 member 10.2.2.2 encapsulation mpls forward permit l2protocol all ! bridge-domain 10 member vfi VPLS-10 member GigabitEthernet2/0/5 service-instance 10 ! interface GigabitEthernet2/0/5 service instance 10 ethernet encapsulation dot1q 10 ! spanning-tree mode mst spanning-tree extend system-id ! spanning-tree mst configuration name myMstName revision 10 instance 1 vlan 20 ! spanning-tree mst 1 priority 0

Example: H-VPLS N-PE Redundancy for MPLS Access using the commands associated with the L2VPN Protocol-Based CLIs feature

The figure below shows a configuration that is set up for the H-VPLS N-PE Redundancy with MPLS Access feature. Because there is no option to configure multihoming on access VPLS, the **xconnect** command is used with priority on uPE1.

Figure 38: H-VPLS N-PE Redundancy with MPLS Access Topology



nPE1 Configuration

```
l2vpn vfi context VPLS-10
 vpn id 10
```

```
 member 102.102.102.102 encapsulation mpls
 member 103.103.103.103 encapsulation mpls
!
bridge-domain 10
 member vfi VPLS-10
 member 105.105.105.105 10 encapsulation mpls
```

nPE2 Configuration

```
l2vpn vfi context VPLS-10
 vpn id 10
 member 101.101.101.101 encapsulation mpls
 member 103.103.103.103 encapsulation mpls
!
bridge-domain 10
 member vfi VPLS-10
 member 105.105.105.105 10 encapsulation mpls
```

nPE3 Configuration

```
l2vpn vfi context VPLS-10
 vpn id 10
 member 101.101.101.101 encapsulation mpls
 member 102.102.102.102 encapsulation mpls
!
bridge-domain 10
 member vfi VPLS-10
```

uPE1 Configuration

```
interface GigabitEthernet0/1/0
 service instance 10 ethernet
 encapsulation dot1q 10
!
l2vpn xconnect context XC-10
 member GigabitEthernet0/1/0 service-instance 10
 member 101.101.101.101 10 encapsulation mpls group pwred priority 9
 member 102.102.102.102 10 encapsulation mpls group pwred priority 10
```

Sample Output on uPE1

```
Device# show l2vpn service peer 101.101.101.101 vcid 10

Legend: St=State    XC St=State in the L2VPN Service      Prio=Priority
        UP=Up       DN=Down            AD=Admin Down      IA=Inactive
        SB=Standby  HS=Hot Standby     RV=Recovering      NH=No Hardware
        m=manually selected

  Interface          Group       Encapsulation                  Prio  St  XC St
  ---------          -----       -------------                  ----  --  -----
VPWS name: foo, State: UP
  Eth1/1.1                       Eth1/1.1:100(Eth VLAN)         0     UP  UP
  pw101              blue        102.1.1.1:100(MPLS)            2     UP  UP
  pw102              blue        103.1.1.1:100(MPLS)            5     SB  IA
  pw103              blue        104.1.1.1:100(MPLS)            8     SB  IA
  pw104              blue        105.1.1.1:100(MPLS)            11    SB  IA

Device# show l2vpn service peer 102.102.102.102 vcid 10
```

```
Legend: St=State     XC St=State in the L2VPN Service      Prio=Priority
        UP=Up        DN=Down              AD=Admin Down     IA=Inactive
        SB=Standby   HS=Hot Standby       RV=Recovering     NH=No Hardware
        m=manually selected

  Interface          Group       Encapsulation                  Prio   St   XC St
  ---------          -----       -------------                  ----   --   -----
VPWS name: foo, State: UP
  Eth1/1.1                       Eth1/1.1:100(Eth VLAN)         0      UP   UP
  pw101              blue        102.1.1.1:100(MPLS)            2      UP   UP
  pw102              blue        103.1.1.1:100(MPLS)            5      SB   IA
  pw103              blue        104.1.1.1:100(MPLS)            8      SB   IA
  pw104              blue        105.1.1.1:100(MPLS)            11     SB   IA
```

Additional References for L2VPN VPLS Inter-AS Option B

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference
IP Routing (BGP) commands	Cisco IOS IP Routing: BGP Command Reference
Concepts and tasks related to configuring the VPLS Autodiscovery: BGP Based feature.	*VPLS Autodiscovery BGP Based*
BGP support for the L2VPN address family	*BGP Support for the L2VPN Address Family*
VPLS	"VPLS Overview" section in the *Configuring Multiprotocol Label Switching on the Optical Services Modules* document
L2VPN multisegment pseudowires, MPLS OAM support for L2VPN multisegment pseudowires, MPLS OAM support for L2VPN inter-AS option B	*L2VPN Multisegment Pseudowires*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing standards has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 4360	*BGP Extended Communities Attribute*
RFC 4364	*BGP/MPLS IP Virtual Private Networks (VPNs)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for H-VPLS N-PE Redundancy for QinQ Access

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 29: Feature Information for H-VPLS N-PE Redundancy for QinQ Access

Feature Name	Releases	Feature Information
H-VPLS N-PE Redundancy for QinQ Access	12.2(33)SRC 12.2(50)SY Cisco IOS XE Release 3.8S	The H-VPLS N-PE Redundancy for QinQ Access feature provides the capability to dual-home a given user provider edge (U-PE) device to two network provide edge (N-PE) devices in order to provide protection against link and node failures. In Cisco IOS Release 12.2(33)SRC, this feature was introduced on the Cisco 7600 series routers. In Cisco IOS Release 12.2(50)SY, this feature was integrated. In Cisco IOS XE Release 3.8S, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. The following commands were introduced or modified: **forward permit l2protocol**, **show mpls l2transport vc**.

Glossary

CE device—customer edge device. A device that belongs to a customer network, which connects to a PE device to utilize MPLS VPN network services.

LAN—local-area network. High-speed, low-error data network covering a relatively small geographic area. LANs connect workstations, peripherals, terminals, and other devices in a single building or other geographically limited areas.

MPLS—Multiprotocol Label Switching. A packet-forwarding technology, used in the network core, that applies data link layer labels to tell switching nodes how to forward data, resulting in faster and more scalable forwarding than network layer routing normally can do.

MSTP—Multiple Spanning Tree Protocol. MSTP enables multiple VLANs to be mapped to the same spanning-tree instance, reducing the number of spanning-tree instances needed to support a large number of VLANs.

N-PE—network provider edge device. This device acts as a gateway between the MPLS core and edge domains.

PE device—provider edge device. The PE device is the entry point into the service provider network. The PE device is typically deployed on the edge of the network and is administered by the service provider.

pseudowire—A pseudowire is a virtual connection that, in the context of VPLS, connects two SVIs. It is a mechanism that carries the elements of an emulated service from one PE device to one or more PE devices over a packet switched network (PSN). A pseudowire is bidirectional and consists of a pair of unidirectional MPLS virtual circuits (VCs). A pseudowire can be used to connect a point-to-point circuit.

QinQ—An IEEE 802.1Q VLAN tunnel. A mechanism for constructing multipoint Layer 2 VPN using Ethernet switches.

redundancy—The duplication of devices, services, or connections so that, in the event of a failure, they can perform the work of those that failed.

router—A network layer device that uses one or more metrics to determine the optimal path along which network traffic should be forwarded. Routers forward packets from one network to another based on network layer information.

spanning tree—Loop-free subset of a network topology.

U-PE—user provider edge device. This device connects CE devices to the service.

VFI—virtual forwarding instance. A VFI is a collection of data structures used by the data plane, software-based or hardware-based, to forward packets to one or more VCs.

VLAN—Virtual LAN. Group of devices on one or more LANs that are configured (using management software) so that they can communicate as if they were attached to the same wire, when in fact they are located on a number of different LAN segments.

VPLS—Virtual Private LAN Service. VPLS describes an architecture that delivers Layer 2 service that emulates an Ethernet LAN across a wide-area network (WAN) and inherits the scaling characteristics of a LAN.

VPLS redundancy—Also called N-PE redundancy. Allows U-PEs to be dual-honed (to their N-PEs) in a loop-free topology with MPLS or QinQ as the access or aggregation domain.

VPN—Virtual Private Network. Allows IP traffic to travel securely over public TCP/IP networks and the Internet by encapsulating and encrypting all IP packets. VPN uses a tunnel to encrypt all information at the IP level.

CHAPTER 18

H-VPLS N-PE Redundancy for MPLS Access

The H-VPLS N-PE Redundancy for MPLS Access feature enables two network provider edge (N-PE) devices to provide failover services to a user provider edge (U-PE) device in a hierarchical virtual private LAN service (H-VPLS). Having redundant N-PE devices provides improved stability and reliability against link and node failures.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for H-VPLS N-PE Redundancy for MPLS Access

- Before configuring this feature, configure your hierarchical virtual private LAN service (H-VPLS) network and make sure it is operating correctly.
- To provide faster convergence, you can enable the MPLS Traffic Engineering—Fast Reroute feature in the Multiprotocol Label Switching (MPLS) core.
- Enable the L2VPN Pseudowire Redundancy feature on the user provider edge (U-PE) devices for MPLS access.

Restrictions for H-VPLS N-PE Redundancy for MPLS Access

- This feature cannot be used with the VPLS Autodiscovery feature on pseudowires that attach to user provider edge (U-PE) devices. When you create the virtual private LAN service (VPLS), you can manually create the virtual forwarding interface (VFI).
- You cannot configure more than one pseudowire to carry the bridge protocol data unit (BPDU) information between the network provider edge (N-PE) devices.
- You cannot configure a local loopback address as a neighbor when you configure the H-VPLS N-PE Redundancy feature on N-PE devices.
- Only two N-PE devices can be connected to each U-PE device.

Information About H-VPLS N-PE Redundancy for MPLS Access

How H-VPLS N-PE Redundancy for MPLS Access

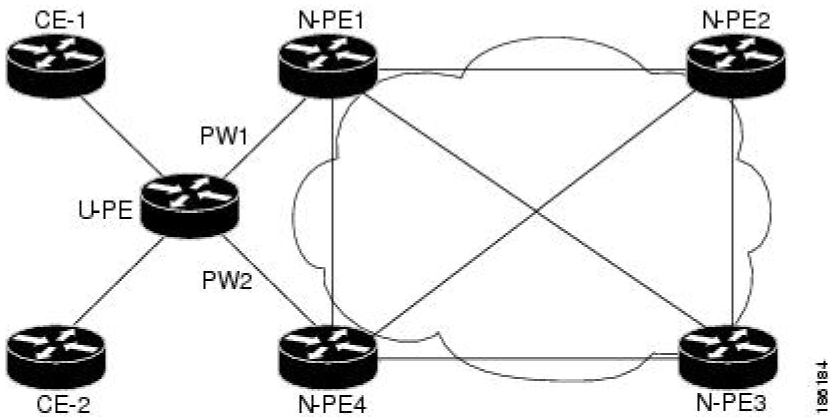
In a network configured with the H-VPLS N-PE Redundancy feature, the user provider edge (U-PE) device is connected to two network provider edge (N-PE) devices. This feature provides a level of redundancy that can tolerate both link and device faults. If a failure occurs in the network that disables one N-PE device from transmitting data, the other N-PE device takes over.

H-VPLS N-PE Redundancy with MPLS Access Based on Pseudowire Redundancy

For the H-VPLS Redundancy with MPLS Access feature based on pseudowire redundancy, the Multiprotocol Label Switching (MPLS) network has pseudowires to the virtual private LAN service (VPLS) core network provider edge (N-PE) devices.

As shown in the figure below, one pseudowire transports data between the user provider edge (U-PE) device and its peer N-PE devices. When a failure occurs along the path of the U-PE device, the backup pseudowire and the redundant N-PE device become active and start transporting data.

Figure 39: H-VPLS N-PE Redundancy for MPLS Access Based on Pseudowire Redundancy



How to Configure H-VPLS N-PE Redundancy for MPLS Access

Specifying the Devices in the Layer 2 VPN VFI

Repeat this task on each N-PE device that is part of the pseudowire redundancy.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *name*
4. **vpn id** *vpn id*
5. **member** *ip-address* **encapsulation mpls**
6. **exit**
7. **bridge-domain** *bridge-id*
8. **member vfi** *vfi-name*
9. **member** *ip-address* [*vc-id*] **encapsulation mpls**
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**l2vpn vfi context** *name* **Example:** `Device(config)# l2vpn vfi context VPLS-10`	Establishes a L2VPN VFI between two or more separate networks, and enters L2VFI configuration mode.
Step 4	**vpn id** *vpn id* **Example:** `Device(config-vfi)# vpn id 10`	Sets a VPN ID on the Virtual Private LAN Services (VPLS) instance. • Use the same VPN ID for the PE devices that belong to the same VPN. • Make sure the VPN ID is unique for each VPN in the service provider network. The range is from 1 to 4294967295.
Step 5	**member** *ip-address* **encapsulation mpls** **Example:** `Device(config-vfi)# member 102.102.102.102 encapsulation mpls`	Specifies the device that forms a point-to-point L2VPN VFI connection. • *ip-address*—IP address of the VFI neighbor (the N-PE device). • **encapsulation mpls**—Specifies Multiprotocol Label Switching (MPLS) as the data encapsulation method.
Step 6	**exit** **Example:** `Device(config-vfi)# exit`	Returns to global configuration mode.
Step 7	**bridge-domain** *bridge-id* **Example:** `Device(config)# bridge-domain 10`	Configures components on a bridge domain, and enters bridge-domain configuration mode.
Step 8	**member vfi** *vfi-name* **Example:** `Device(config-bdomain)# member vfi VPLS-10`	Configures the VFI member in the bridge-domain.
Step 9	**member** *ip-address* [*vc-id*] **encapsulation mpls** **Example:** `Device(config-vfi)# member 105.105.105.105 10 encapsulation mpls`	Specifies the device that forms a point-to-point Layer 2 VPN (L2VPN) VFI connection. • *ip-address*—IP address of the VFI neighbor (U-PE device). • *vc-id*—Virtual circuit identifier. • **encapsulation mpls**—Specifies MPLS as the data encapsulation method.
Step 10	**end** **Example:** `Device(config-bdomain)# end`	Returns to privileged EXEC mode.

Specifying the N-PE Devices That Form the Layer 2 VPN Cross Connection With the U-PE

Perform this task on the U-PE device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **exit**
7. **exit**
8. **l2vpn xconnect context** *context-name*
9. **member gigabitethernet** *interface-number* [**service-instance** *id*]
10. **member** *ip-address vc-id* **encapsulation mpls** [**group** *group-name* [**priority** *number*]]
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet0/1/0`	Specifies the interface to configure, and enters interface configuration mode.
Step 4	**service instance** *id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Configures an Ethernet service instance on theinterface, and enters Ethernet service configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 10`	Defines the matching criteria to map 802.1Q frames ingress on the interface to the appropriate service instance.
Step 6	**exit** **Example:** `Device(config-if-srv)# exit`	Returns to interface configuration mode.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 8	**l2vpn xconnect context** *context-name* **Example:** `Device(config)# l2vpn xconnect context XC-10`	Creates a Layer 2 VPN (L2VPN) cross-connect context, and enters xconnect configuration mode.
Step 9	**member gigabitethernet** *interface-number* [**service-instance** *id*] **Example:** `Device(config-xconnect)# member GigabitEthernet0/1/0 service-instance 10`	Specifies devices that form a Layer 2 VPN (L2VPN) cross connect. • **service-instance** *id*—(Optional) Specifies the service instance identifier.

Step 10	**member** *ip-address vc-id* **encapsulation mpls** [**group** *group-name* [**priority** *number*]]
	Example:
	`Device(config-xconnect)# member 101.101.101.101 10 encapsulation mpls group pwred priority 9`
	`Device(config-xconnect)# member 102.102.102.102 10 encapsulation mpls group pwred priority 10`
		Specifies devices that form a Layer 2 VPN (L2VPN) cross connect.
		• *ip-address*—IP address of the peer N-PE device.
		• *vc-id*—Virtual circuit identifier.
		• **encapsulation mpls**—Specifies Multiprotocol Label Switching (MPLS) as the data encapsulation method.
		• **group** *group-name*—Specifies the cross-connect member redundancy group name.
		• **priority** *number*—Specifies the cross-connect member priority. The range is from 0 to 16. The highest priority is 0. Lowest priority is 16.

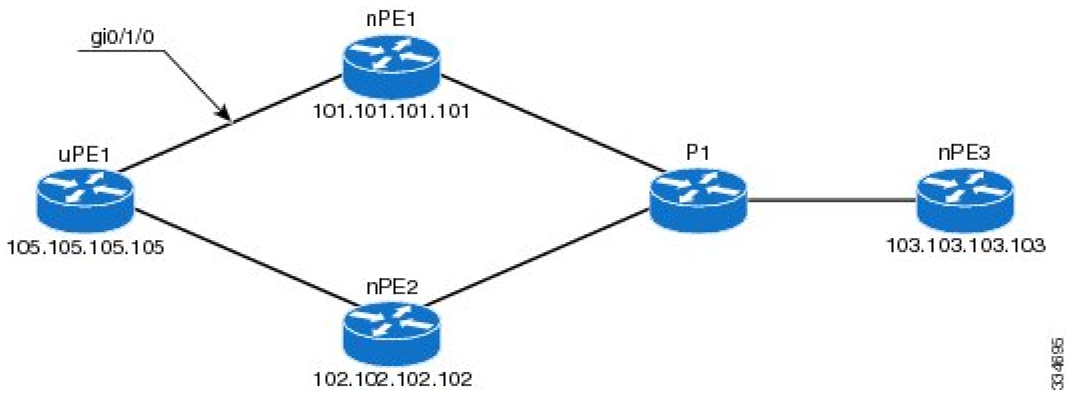
Step 11	**end** **Example:** `Device(config-xconnect)# end`	Returns to privileged EXEC mode.

Configuration Examples for H-VPLS N-PE Redundancy for MPLS Access

Example: H-VPLS N-PE Redundancy for MPLS Access

The figure below shows a configuration that is set up for the H-VPLS N-PE Redundancy with MPLS Access feature. Since there is no option to configure multihoming on access VPLS, the **xconnect** command is used with priority on uPE1. Please let me know if you need any other info.

Figure 40: H-VPLS N-PE Redundancy with MPLS Access Topology



nPE1 Configuration

```
l2vpn vfi context VPLS-10
 vpn id 10
 member 102.102.102.102 encapsulation mpls
 member 103.103.103.103 encapsulation mpls
!
bridge-domain 10
 member vfi VPLS-10
 member 105.105.105.105 10 encapsulation mpls
```

nPE2 Configuration

```
l2vpn vfi context VPLS-10
 vpn id 10
 member 101.101.101.101 encapsulation mpls
 member 103.103.103.103 encapsulation mpls
!
bridge-domain 10
 member vfi VPLS-10
 member 105.105.105.105 10 encapsulation mpls
```

nPE3 Configuration

```
l2vpn vfi context VPLS-10
 vpn id 10
 member 101.101.101.101 encapsulation mpls
 member 102.102.102.102 encapsulation mpls
!
bridge-domain 10
 member vfi VPLS-10
```

uPE1 Configuration

```
interface GigabitEthernet0/1/0
 service instance 10 ethernet
 encapsulation dot1q 10
!
l2vpn xconnect context XC-10
 member GigabitEthernet0/1/0 service-instance 10
 member 101.101.101.101 10 encapsulation mpls group pwred priority 9
 member 102.102.102.102 10 encapsulation mpls group pwred priority 10
```

Sample Output on uPE1

```
Device# show xconnect peer 101.101.101.101 vcid 10

Legend:    XC ST=Xconnect State  S1=Segment1 State  S2=Segment2 State
  UP=Up       DN=Down            AD=Admin Down      IA=Inactive
  SB=Standby  HS=Hot Standby     RV=Recovering      NH=No Hardware

XC ST  Segment 1                         S1 Segment 2                          S2
------+---------------------------------+--+----------------------------------+--
UP pri   ac Gi0/1/0:10(Eth VLAN)         UP mpls 101.101.101.101:10           UP

Device# show xconnect peer 102.102.102.102 vcid 10

Legend:    XC ST=Xconnect State  S1=Segment1 State  S2=Segment2 State
  UP=Up       DN=Down            AD=Admin Down      IA=Inactive
  SB=Standby  HS=Hot Standby     RV=Recovering      NH=No Hardware

XC ST  Segment 1                         S1 Segment 2                          S2
------+---------------------------------+--+----------------------------------+--
IA pri   ac Gi0/1/0:10(Eth VLAN)         UP mpls 102.102.102.102:10           SB
Device#
```

Additional References for L2VPN VPLS Inter-AS Option B

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference
IP Routing (BGP) commands	Cisco IOS IP Routing: BGP Command Reference
Concepts and tasks related to configuring the VPLS Autodiscovery: BGP Based feature.	*VPLS Autodiscovery BGP Based*
BGP support for the L2VPN address family	*BGP Support for the L2VPN Address Family*
VPLS	"VPLS Overview" section in the *Configuring Multiprotocol Label Switching on the Optical Services Modules* document
L2VPN multisegment pseudowires, MPLS OAM support for L2VPN multisegment pseudowires, MPLS OAM support for L2VPN inter-AS option B	*L2VPN Multisegment Pseudowires*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing standards has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 4360	*BGP Extended Communities Attribute*
RFC 4364	*BGP/MPLS IP Virtual Private Networks (VPNs)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for H-VPLS N-PE Redundancy for MPLS Access

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 30: Feature Information for H-VPLS N-PE Redundancy for MPLS Access

Feature Name	Releases	Feature Information
H-VPLS N-PE Redundancy for MPLS Access	Cisco IOS XE Release 3.6S	The H-VPLS N-PE Redundancy for MPLS Access feature enables two network provider edge (N-PE) devices to provide redundancy to a user provider edge (U-PE) device in a hierarchical virtual private LAN service (H-VPLS). Having redundant N-PE devices provides improved stability and reliability against link and node failures. In Cisco IOS XE Release 3.6S, support was added for the Cisco ASR 903 Router. The following commands were introduced or modified: **forward permit l2protocol**, **show mpls l2transport vc**.

Glossary

CE device—customer edge device. A device that belongs to a customer network, which connects to a PE device to utilize MPLS VPN network services.

LAN—local-area network. High-speed, low-error data network covering a relatively small geographic area. LANs connect workstations, peripherals, terminals, and other devices in a single building or other geographically limited areas.

MPLS—Multiprotocol Label Switching. A packet-forwarding technology, used in the network core, that applies data link layer labels to tell switching nodes how to forward data, resulting in faster and more scalable forwarding than network layer routing normally can do.

MSTP—Multiple Spanning Tree Protocol. MSTP enables multiple VLANs to be mapped to the same spanning-tree instance, reducing the number of spanning-tree instances needed to support a large number of VLANs.

N-PE—network provider edge device. This device acts as a gateway between the MPLS core and edge domains.

PE device—provider edge device. The PE device is the entry point into the service provider network. The PE device is typically deployed on the edge of the network and is administered by the service provider.

pseudowire—A pseudowire is a virtual connection that, in the context of VPLS, connects two SVIs. It is a mechanism that carries the elements of an emulated service from one PE device to one or more PE devices over a packet switched network (PSN). A pseudowire is bidirectional and consists of a pair of unidirectional MPLS virtual circuits (VCs). A pseudowire can be used to connect a point-to-point circuit.

QinQ—An IEEE 802.1Q VLAN tunnel. A mechanism for constructing multipoint Layer 2 VPN using Ethernet switches.

redundancy—The duplication of devices, services, or connections so that, in the event of a failure, they can perform the work of those that failed.

router—A network layer device that uses one or more metrics to determine the optimal path along which network traffic should be forwarded. Routers forward packets from one network to another based on network layer information.

spanning tree—Loop-free subset of a network topology.

U-PE—user provider edge device. This device connects CE devices to the service.

VFI—virtual forwarding instance. A VFI is a collection of data structures used by the data plane, software-based or hardware-based, to forward packets to one or more VCs.

VLAN—Virtual LAN. Group of devices on one or more LANs that are configured (using management software) so that they can communicate as if they were attached to the same wire, when in fact they are located on a number of different LAN segments.

VPLS—Virtual Private LAN Service. VPLS describes an architecture that delivers Layer 2 service that emulates an Ethernet LAN across a wide-area network (WAN) and inherits the scaling characteristics of a LAN.

VPLS redundancy—Also called N-PE redundancy. Allows U-PEs to be dual-honed (to their N-PEs) in a loop-free topology with MPLS or QinQ as the access or aggregation domain.

VPN—Virtual Private Network. Allows IP traffic to travel securely over public TCP/IP networks and the Internet by encapsulating and encrypting all IP packets. VPN uses a tunnel to encrypt all information at the IP level.

CHAPTER 19

VPLS MAC Address Withdrawal

The VPLS MAC Address Withdrawal feature provides faster convergence by removing (or unlearning) MAC addresses that have been dynamically learned. A Label Distribution Protocol (LDP)-based MAC address withdrawal message is used for this purpose. A MAC list Type Length Value (TLV) is part of the MAC address withdrawal message. No configuration is needed.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About VPLS MAC Address Withdrawal

VPLS MAC Address Withdrawal

The VPLS MAC Address Withdrawal feature provides faster convergence by removing (or unlearning) MAC addresses that have been dynamically learned. A Label Distribution Protocol (LDP)-based MAC address withdrawal message is used for this purpose. A MAC list Type Length Value (TLV) is part of the MAC address withdrawal message.

The **debug mpls ldp messages** and **debug mpls ldp session io** commands support monitoring of MAC address withdrawal messages being exchanged between LDP peers. Any Transport over Multiprotocol Label Switching (AToM) might provide other means to display or monitor MAC address withdrawal messages. The Tag Distribution Protocol (TDP) is not supported because AToM uses only LDP for the MAC address withdrawal message.

PE devices learn the remote MAC addresses and directly attached MAC addresses on customer-facing ports by deriving the topology and forwarding information from packets originating at customer sites. To display the number of MAC address withdrawal messages, enter the **show mpls l2transport vc detail** command, as shown in the following example:

```
Device# show mpls l2transport vc detail

Local interface: VFI TEST VFI up
  MPLS VC type is VFI, interworking type is Ethernet
  Destination address: 10.1.1.1, VC ID: 1000, VC status: up
    Output interface: Se2/0, imposed label stack {17}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:04:34, last status change time: 00:04:15
  Signaling protocol: LDP, peer 10.1.1.1:0 up
    Targeted Hello: 10.1.1.1(LDP Id) -> 10.1.1.1
    MPLS VC labels: local 16, remote 17
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
    MAC Withdraw: sent 5, received 3
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 0, send 0
    byte totals:   receive 0, send 0
    packet drops:  receive 0, send 0
```

VPLS MAC Address Withdrawal Using Commands Associated with L2VPN Protocol-Based Feature

The VPLS MAC Address Withdrawal feature provides faster convergence by removing (or unlearning) MAC addresses that have been dynamically learned. A Label Distribution Protocol (LDP)-based MAC address withdrawal message is used for this purpose. A MAC list Type Length Value (TLV) is part of the MAC address withdrawal message.

The **debug mpls ldp messages** and **debug mpls ldp session io** commands support monitoring of MAC address withdrawal messages being exchanged between LDP peers. Any Transport over Multiprotocol Label Switching (AToM) might provide other means to display or monitor MAC address withdrawal messages. The Tag Distribution Protocol (TDP) is not supported because AToM uses only LDP for the MAC address withdrawal message.

PE devices learn the remote MAC addresses and directly attached MAC addresses on customer-facing ports by deriving the topology and forwarding information from packets originating at customer sites. To display the number of MAC address withdrawal messages, enter the **show l2vpn atom vc detail** command, as shown in the following example:

```
Device# show l2vpn atom vc detail

Local interface: VFI TEST VFI up
  MPLS VC type is VFI, interworking type is Ethernet
  Destination address: 10.1.1.1, VC ID: 1000, VC status: up
    Output interface: Se2/0, imposed label stack {17}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:04:34, last status change time: 00:04:15
```

```
Signaling protocol: LDP, peer 10.1.1.1:0 up
  Targeted Hello: 10.1.1.1(LDP Id) -> 10.1.1.1
  MPLS VC labels: local 16, remote 17
  Group ID: local 0, remote 0
  MTU: local 1500, remote 1500
  Remote interface description:
  MAC Withdraw: sent 5, received 3
Sequencing: receive disabled, send disabled
VC statistics:
  packet totals: receive 0, send 0
  byte totals:   receive 0, send 0
  packet drops:  receive 0, send 0
```

How MAC Address Withdrawal Works with H-VPLS N-PE Redundancy with MPLS Access

If the pseudowire between the user provider edge (U-PE) device and network provider edge (N-PE) device fails, the L2VPN Pseudowire Redundancy feature on the U-PE device activates the standby pseudowire. In addition, the U-PE device sends a Label Distribution Protocol (LDP) MAC address withdrawal request to the new N-PE device, which forwards the message to all pseudowires in the virtual private LAN service (VPLS) core and flushes its MAC address table.

If a on the N-PE device fails, the L2VPN Pseudowire Redundancy feature activates the standby pseudowire and the U-PE device sends a MAC withdrawal message to the newly active N-PE device.

How MAC Address Withdrawal Works with H-VPLS N-PE Redundancy with QinQ Access

If a failure occurs in the customer-switched network, a spanning-tree Topology Change Notification (TCN) is issued to the network provider edge (N-PE) device, which issues a Label Distribution Protocol (LDP)-based MAC address withdrawal message to the peer N-PE devices and flushes its MAC address table.

Additional References for Any Transport over MPLS

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for VPLS MAC Address Withdrawal

Table 31: Feature Information for VPLS MAC Address Withdrawal

Feature Name	Releases	Feature Information
VPLS MAC Address Withdrawal	Cisco IOS XE Release 3.5S	The VPLS MAC Address Withdrawal feature provides faster convergence by removing (or unlearning) MAC addresses that have been dynamically learned. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 Router. No commands were introduced or modified.

CHAPTER 20

Configuring Virtual Private LAN Services

Virtual Private LAN Services (VPLS) enables enterprises to link together their Ethernet-based LANs from multiple sites via the infrastructure provided by their service provider.

This module explains VPLS and how to configure it.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Virtual Private LAN Services

Before you configure Virtual Private LAN Services (VPLS), ensure that the network is configured as follows:

- Configure IP routing in the core so that provider edge (PE) devices can reach each other via IP.
- Configure Multiprotocol Label Switching (MPLS) in the core so that a label switched path (LSP) exists between PE devices.
- Configure a loopback interface for originating and terminating Layer 2 traffic. Ensure that PE devices can access the loopback interface of the other device. Note that the loopback interface is not required in all cases. For example, tunnel selection does not need a loopback interface when VPLS is directly mapped to a traffic engineering (TE) tunnel.

- Identify peer PE devices and attach Layer 2 circuits to VPLS at each PE device.

Restrictions for Virtual Private LAN Services

The following general restrictions apply to all transport types under Virtual Private LAN Services (VPLS):

- If you do not enable the EFP feature template, then there is no traffic flow between EFP and VFI (when EFP is with Split Horizon group and VFI is default). But when you enable the EFP feature template, then there is traffic flow between EFP and VFI because of design limitations.
- Supported maximum values:
    - Total number of virtual forwarding instances (VFIs): 4096 (4 K)
- Software-based data plane is not supported.
- Load sharing and failover on redundant customer-edge-provider-edge (CE-PE) links are not supported.

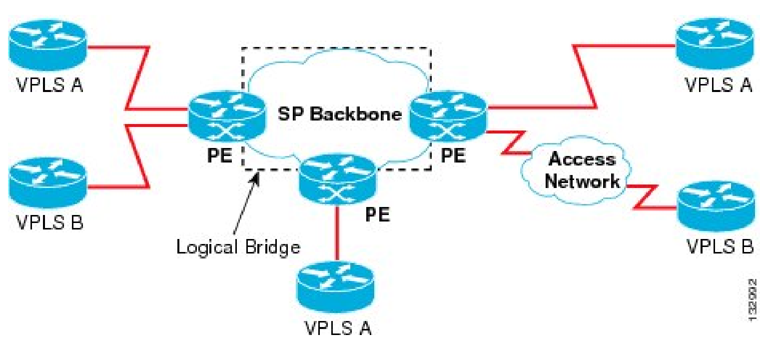
Information About Virtual Private LAN Services

VPLS Overview

Virtual Private LAN Services (VPLS) enables enterprises to link together their Ethernet-based LANs from multiple sites via the infrastructure provided by their service provider. From the enterprise perspective, the service provider's public network looks like one giant Ethernet LAN. For the service provider, VPLS provides an opportunity to deploy another revenue-generating service on top of the existing network without major capital expenditures. Operators can extend the operational life of equipment in their network.

VPLS uses the provider core to join multiple attachment circuits together to simulate a virtual bridge that connects the multiple attachment circuits together. From a customer point of view, there is no topology for VPLS. All customer edge (CE) devices appear to connect to a logical bridge emulated by the provider core (see the figure below).

Figure 41: VPLS Topology



Full-Mesh Configuration

A full-mesh configuration requires a full mesh of tunnel label switched paths (LSPs) between all provider edge (PE) devices that participate in Virtual Private LAN Services (VPLS). With a full mesh, signaling overhead and packet replication requirements for each provisioned virtual circuit (VC) on a PE can be high.

You set up a VPLS by first creating a virtual forwarding instance (VFI) on each participating PE device. The VFI specifies the VPN ID of a VPLS domain, the addresses of other PE devices in the domain, and the type of tunnel signaling and encapsulation mechanism for each peer PE device.

The set of VFIs formed by the interconnection of the emulated VCs is called a VPLS instance; it is the VPLS instance that forms the logic bridge over a packet switched network. After the VFI has been defined, it needs to be bound to an attachment circuit to the CE device. The VPLS instance is assigned a unique VPN ID.

PE devices use the VFI to establish a full-mesh LSP of emulated VCs to all other PE devices in the VPLS instance. PE devices obtain the membership of a VPLS instance through static configuration using the Cisco IOS CLI.

A full-mesh configuration allows the PE device to maintain a single broadcast domain. When the PE device receives a broadcast, multicast, or unknown unicast packet on an attachment circuit (AC), it sends the packet out on all other ACs and emulated circuits to all other CE devices participating in that VPLS instance. The CE devices see the VPLS instance as an emulated LAN.

To avoid the problem of a packet looping in the provider core, PE devices enforce a “split-horizon” principle for emulated VCs. In a split horizon, if a packet is received on an emulated VC, it is not forwarded on any other emulated VC.

The packet forwarding decision is made by looking up the Layer 2 VFI of a particular VPLS domain.

A VPLS instance on a particular PE device receives Ethernet frames that enter on specific physical or logical ports and populates a MAC table similarly to how an Ethernet switch works. The PE device can use the MAC address to switch these frames into the appropriate LSP for delivery to the another PE device at a remote site.

If the MAC address is not available in the MAC address table, the PE device replicates the Ethernet frame and floods it to all logical ports associated with that VPLS instance, except the ingress port from which it just entered. The PE device updates the MAC table as it receives packets on specific ports and removes addresses not used for specific periods.

Static VPLS Configuration

Virtual Private LAN Services (VPLS) over Multiprotocol Label Switching-Transport Profile (MPLS-TP) tunnels allows you to deploy a multipoint-to-multipoint layer 2 operating environment over an MPLS-TP network for services such as Ethernet connectivity and multicast video. To configure static VPLS, you must specify a static range of MPLS labels using the **mpls label range** command with the **static** keyword.

H-VPLS

Hierarchical VPLS (H-VPLS) reduces signaling and replication overhead by using full-mesh and hub-and-spoke configurations. Hub-and-spoke configurations operate with split horizon to allow packets to be switched between pseudowires (PWs), effectively reducing the number of PWs between provider edge (PE) devices.

Note Split horizon is the default configuration to avoid broadcast packet looping.

Supported Features

Multipoint-to-Multipoint Support

In a multipoint-to-multipoint network, two or more devices are associated over the core network. No single device is designated as the Root node; all devices are considered as Root nodes. All frames can be exchanged directly between the nodes.

Non-Transparent Operation

A virtual Ethernet connection (VEC) can be transparent or non-transparent with respect to Ethernet protocol data units (PDUs). The VEC non-transparency allows users to have a Frame Relay-type service between Layer 3 devices.

Circuit Multiplexing

Circuit multiplexing allows a node to participate in multiple services over a single Ethernet connection. By participating in multiple services, the Ethernet connection is attached to multiple logical networks. Some examples of possible service offerings are VPN services between sites, Internet services, and third-party connectivity for intercompany communications.

MAC-Address Learning, Forwarding, and Aging

Provider edge (PE) devices must learn remote MAC addresses and directly attached MAC addresses on ports that face the external network. MAC address learning accomplishes this by deriving the topology and forwarding information from packets originating at customer sites. A timer is associated with stored MAC addresses. After the timer expires, the entry is removed from the table.

Jumbo Frame Support

Jumbo frame support provides support for frame sizes between 1548 and 9216 bytes. You use the CLI to establish the jumbo frame size for any value specified in the above range. The default value is 1500 bytes in any Layer 2/VLAN interface. You can configure jumbo frame support on a per-interface basis.

Q-in-Q Support and Q-in-Q to EoMPLS Support

With 802.1Q tunneling (Q-in-Q), the customer edge (CE) device issues VLAN-tagged packets and VPLS forwards these packets to a far-end CE device. Q-in-Q refers to the fact that one or more 802.1Q tags may be located in a packet within the interior of the network. As packets are received from a CE device, an additional VLAN tag is added to incoming Ethernet packets to segregate traffic from different CE devices. Untagged packets originating from a CE device use a single tag within the interior of the VLAN switched network, whereas previously tagged packets originating from the CE device use two or more tags.

VPLS Services

Transparent LAN Service

Transparent LAN Service (TLS) is an extension to the point-to-point port-based Ethernet over Multiprotocol Label Switching (EoMPLS), which provides bridging protocol transparency (for example, bridge protocol data units [BPDUs]) and VLAN values. Bridges see this service as an Ethernet segment. With TLS, the PE device forwards all Ethernet packets received from the customer-facing interface (including tagged and untagged packets, and BPDUs) as follows:

- To a local Ethernet interface or an emulated virtual circuit (VC) if the destination MAC address is found in the Layer 2 forwarding table.
- To all other local Ethernet interfaces and emulated VCs belonging to the same VPLS domain if the destination MAC address is a multicast or broadcast address or if the destination MAC address is not found in the Layer 2 forwarding table.

Note You must enable Layer 2 protocol tunneling to run the Cisco Discovery Protocol (CDP), the VLAN Trunking Protocol (VTP), and the Spanning-Tree Protocol (STP).

Ethernet Virtual Connection Service

Ethernet Virtual Connection Service (EVCS) is an extension to the point-to-point VLAN-based Ethernet over MPLS (EoMPLS) that allows devices to reach multiple intranet and extranet locations from a single physical port. With EVCS, the provider edge (PE) device forwards all Ethernet packets with a particular VLAN tag received from the customer-facing interface (excluding bridge protocol data units [BPDUs]) as follows:

- To a local Ethernet interface or to an emulated virtual circuit (VC) if the destination MAC address is found in the Layer 2 forwarding table.
- To all other local Ethernet interfaces and emulated VCs belonging to the same Virtual Private LAN Services (VPLS) domain if the destination MAC address is a multicast or a broadcast address or if the destination MAC address is not found in the Layer 2 forwarding table.

Note Because it has only local significance, the demultiplexing VLAN tag that identifies a VPLS domain is removed before the packet is forwarded to the outgoing Ethernet interfaces or emulated VCs.

VPLS Integrated Routing and Bridging

Virtual Private LAN Services (VPLS) integrated routing and bridging routes Layer 3 traffic and switches Layer 2 frames for pseudowire connections between provider edge (PE) devices using a VPLS multipoint PE device. The ability to route frames to and from these interfaces supports the termination of a pseudowire into a Layer 3 network (VPN or global) on the same switch or to tunnel Layer 3 frames over a Layer 2 tunnel (VPLS).

To configure routing support for a pseudowire, configure an IP address and other Layer 3 features for the Layer 3 domain in interface configuration mode.

Note VPLS integrated routing and bridging does not support multicast routing. VPLS integrated routing and bridging is also known as routed pseudowire and routed VPLS.

The following example shows how to assign IP address 10.10.10.1 to a bridge domain interface (BDI).

```
interface bdi 100
  ip address 10.10.10.1 255.255.255.0
```

VPLS and Type 4 dummy VLAN Tag

From Cisco IOS XE Everest 16.4.1 release, VPLS VC type 4 mode (with autodiscovery) can be used to configure a dummy VLAN tag. This feature can be used to modify the VLAN ID to filter based on the VLAN ID. The dummy VLAN ID is 0 in default VPLS type 4 mode, and can be set to any value from 1 to 4094. Refer to the section titled *"Example: MAC ACL with Dummy VLAN ID"* in this chapter for the configuration example.

How to Configure Virtual Private LAN Services

Provisioning a Virtual Private LAN Services (VPLS) link involves provisioning the associated attachment circuit and a virtual forwarding instance (VFI) on a provider edge (PE) device.

In Cisco IOS XE Release 3.7S, the L2VPN Protocol-Based CLIs feature was introduced. This feature provides a set of processes and an improved infrastructure for developing and delivering Cisco IOS software on various Cisco platforms. This feature introduces new commands and modifies or replaces existing commands to achieve a consistent functionality across Cisco platforms and provide cross-Operating System (OS) support.

This section consists of tasks that use the commands existing prior to Cisco IOS XE Release 3.7S and a corresponding task that uses the commands introduced or modified by the L2VPN Protocol-Based CLIs feature.

Configuring PE Layer 2 Interfaces on CE Devices

You can configure the Ethernet flow point (EFP) as a Layer 2 virtual interface. You can also select tagged or untagged traffic from a customer edge (CE) device.

Configuring 802.1Q Access Ports for Tagged Traffic from a CE Device

Note When Ethernet Virtual Connection Service (EVCS) is configured, a provider edge (PE) device forwards all Ethernet packets with a particular VLAN tag to a local Ethernet interface or emulated virtual circuit (VC) if the destination MAC address is found in the Layer 2 forwarding table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **negotiation auto**
6. **service instance** *si-id* **ethernet**
7. **encapsulation dot1q** *vlan-id*
8. **bridge-domain** *bd-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.
Step 6	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Specifies the service instance ID and enters service instance configuration mode.
Step 7	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 200`	Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance. Ensure that the interface on the adjoining customer edge (CE) device is on the same VLAN as this PE device.
Step 8	**bridge-domain** *bd-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 9	**end** **Example:** `Device(config-if-srv)# end`	Exits service instance configuration mode and returns to privileged EXEC mode.

Configuring 802.1Q Access Ports for Tagged Traffic from a CE Device: Alternate Configuration

Note When Ethernet Virtual Connection Service (EVCS) is configured, the PE device forwards all Ethernet packets with a particular VLAN tag to a local Ethernet interface or an emulated virtual circuit (VC) if the destination MAC address is found in the Layer 2 forwarding table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **negotiation auto**
6. **service instance** *si-id* **ethernet**
7. **encapsulation dot1q** *vlan-id*
8. **exit**
9. **exit**
10. **bridge-domain** *bd-id*
11. **member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*]
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.

	Command or Action	Purpose
Step 5	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.
Step 6	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Specifies a service instance ID and enters service instance configuration mode.
Step 7	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 200`	Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance. • Ensure that the interface on the adjoining customer edge (CE) device is on the same VLAN as this provider edge (PE) device.
Step 8	**exit** **Example:** `Device(config-if-srv)# exit`	Exits service instance configuration mode and returns to interface configuration mode.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**bridge-domain** *bd-id* **Example:** `Device(config)# bridge-domain 100`	Specifies the bridge domain ID and enters bridge-domain configuration mode.
Step 11	**member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*] **Example:** `Device(config-bdomain)# member gigabitethernet0/0/1 service-instance 1000`	Binds a service instance to a bridge domain instance.
Step 12	**end** **Example:** `Device(config-bdomain)# end`	Exits bridge-domain configuration mode and returns to privileged EXEC mode.

Configuring Access Ports for Untagged Traffic from a CE Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **negotiation auto**
6. **service instance** *si-id* **ethernet**
7. **encapsulation untagged**
8. **bridge-domain** *bd-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Specifies an interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.
Step 6	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Specifies a service instance ID and enters service instance configuration mode.

	Command or Action	Purpose
Step 7	**encapsulation untagged** **Example:** `Device(config-if-srv)# encapsulation untagged`	Defines the matching criteria to map untagged ingress Ethernet frames on an interface to the appropriate service instance. • Ensure that the interface on the adjoining customer edge (CE) device is on the same VLAN as this provider edge (PE) device.
Step 8	**bridge-domain** *bd-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance or MAC tunnel to a bridge domain instance.
Step 9	**end** **Example:** `Device(config-if-srv)# end`	Exits service instance configuration mode and returns to privileged EXEC mode.

Configuring Access Ports for Untagged Traffic from a CE Device: Alternate Configuration

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **negotiation auto**
6. **service instance** *si-id* **ethernet**
7. **encapsulation untagged**
8. **exit**
9. **exit**
10. **bridge-domain** *bd-id*
11. **member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*]
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/4/4`	Specifies an interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.
Step 6	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Specifies a service instance ID and enters service instance configuration mode.
Step 7	**encapsulation untagged** **Example:** `Device(config-if-srv)# encapsulation untagged`	Defines the matching criteria to map untagged ingress Ethernet frames on an interface to the appropriate service instance. • Ensure that the interface on the adjoining customer edge (CE) device is on the same VLAN as this provider edge (PE) device.
Step 8	**exit** **Example:** `Device(config-if-srv)# exit`	Exits service instance configuration mode and returns to interface configuration mode.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**bridge-domain** *bd-id* **Example:** `Device(config)# bridge-domain 100`	Specifies the bridge domain ID and enters bridge-domain configuration mode.
Step 11	**member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*]	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose
	Example: `Device(config-bdomain)# member gigabitethernet0/4/4 service-instance 1000`	
Step 12	**end** **Example:** `Device(config-bdomain)# end`	Exits bridge-domain configuration mode and returns to privileged EXEC mode.

Configuring Q-in-Q EFP

Note When a thread-local storage (TLS) is configured, the provider edge (PE) device forwards all Ethernet packets received from the customer edge (CE) device to all local Ethernet interfaces and emulated virtual circuits (VCs) that belong to the same Virtual Private LAN Services (VPLS) domain if the MAC address is not found in the Layer 2 forwarding table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **negotiation auto**
6. **service instance** *si-id* **ethernet**
7. **encapsulation dot1q** *vlan-id* **second-dot1q** *vlan-id*
8. **bridge-domain** *bd-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 0/0/2`	
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.
Step 6	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Specifies a service instance ID and enters service instance configuration mode.
Step 7	**encapsulation dot1q** *vlan-id* **second-dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 200 second-dot1q 400`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance. • Ensure that the interface on the adjoining CE device is on the same VLAN as this PE device.
Step 8	**bridge-domain** *bd-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance or a MAC tunnel to a bridge domain instance.
Step 9	**end** **Example:** `Device(config-if-srv)# end`	Exits service instance configuration mode and returns to privileged EXEC mode.

Configuring Q-in-Q EFP: Alternate Configuration

Note When a thread-local storage (TLS) is configured, the provider edge (PE) device forwards all Ethernet packets received from the customer edge (CE) device to all local Ethernet interfaces and emulated virtual circuits (VCs) belonging to the same Virtual Private LAN Services (VPLS) domain if the MAC address is not found in the Layer 2 forwarding table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*

4. **no ip address** [*ip-address mask*] [**secondary**]
5. **negotiation auto**
6. **service instance** *si-id* **ethernet**
7. **encapsulation dot1q** *vlan-id* **second-dot1q** *vlan-id*
8. **exit**
9. **exit**
10. **bridge-domain** *bd-id*
11. **member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*]
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/2`	Specifies an interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.
Step 6	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Specifies a service instance ID and enters service instance configuration mode.
Step 7	**encapsulation dot1q** *vlan-id* **second-dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 200 second-dot1q 400`	Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance. • Ensure that the interface on the adjoining CE device is on the same VLAN as this PE device.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config-if-srv)# exit`	Exits service instance configuration mode and returns to interface configuration mode.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**bridge-domain** *bd-id* **Example:** `Device(config)# bridge-domain 100`	Specifies the bridge domain ID and enters bridge-domain configuration mode.
Step 11	**member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*] **Example:** `Device(config-bdomain)# member gigabitethernet0/0/2 service-instance 1000`	Binds a service instance to a bridge domain instance.
Step 12	**end** **Example:** `Device(config-bdomain)# end`	Exits bridge-domain configuration mode and returns to privileged EXEC mode.

Configuring MPLS on a PE Device

To configure Multiprotocol Label Switching (MPLS) on a provider edge (PE) device, configure the required MPLS parameters.

Note Before configuring MPLS, ensure that IP connectivity exists between all PE devices by configuring Interior Gateway Protocol (IGP), Open Shortest Path First (OSPF), or Intermediate System to Intermediate System (IS-IS) between PE devices.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol** {**ldp** | **tdp**}
4. **mpls ldp logging neighbor-changes**
5. **mpls ldp discovery hello holdtime** *seconds*
6. **mpls ldp router-id** *interface-type-number* [**force**]

7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mpls label protocol** {**ldp** | **tdp**} **Example:** `Device(config)# mpls label protocol ldp`	Specifies the label distribution protocol for the platform.
Step 4	**mpls ldp logging neighbor-changes** **Example:** `Device(config)# mpls ldp logging neighbor-changes`	(Optional) Generates system error logging (syslog) messages when LDP sessions go down.
Step 5	**mpls ldp discovery hello holdtime** *seconds* **Example:** `Device(config)# mpls ldp discovery hello holdtime 5`	Configures the interval between the transmission of consecutive LDP discovery hello messages or the hold time for an LDP transport connection.
Step 6	**mpls ldp router-id** *interface-type-number* [**force**] **Example:** `Device(config)# mpls ldp router-id loopback0 force`	Specifies a preferred interface for the LDP router ID.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a VFI on a PE Device

The virtual forwarding interface (VFI) specifies the VPN ID of a Virtual Private LAN Services (VPLS) domain, the addresses of other provider edge (PE) devices in the domain, and the type of tunnel signaling and encapsulation mechanism for each peer.

Note Only Multiprotocol Label Switching (MPLS) encapsulation is supported.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *name* **manual**
4. **vpn id** *vpn-id*
5. **neighbor** *remote-router-id vc-id* {**encapsulation** *encapsulation-type* | **pw-class** *pw-name*} [**no-split-horizon**]
6. **bridge-domain** *bd-id*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *name* **manual** **Example:** `Device(config)# l2 vfi vfi110 manual`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) between two or more separate networks and enters VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 110`	Configures a VPN ID for a VPLS domain. • The emulated VCs bound to this Layer 2 virtual routing and forwarding (VRF) instance use this VPN ID for signaling.
Step 5	**neighbor** *remote-router-id vc-id* {**encapsulation** *encapsulation-type* | **pw-class** *pw-name*} [**no-split-horizon**] **Example:** `Device(config-vfi)# neighbor 172.16.10.2 4 encapsulation mpls`	Specifies the type of tunnel signaling and encapsulation mechanism for each VPLS peer. **Note** Split horizon is the default configuration to avoid broadcast packet looping and to isolate Layer 2 traffic. Use the **no-split-horizon** keyword to disable split horizon and to configure multiple VCs per spoke into the same VFI.

	Command or Action	Purpose
Step 6	**bridge-domain** *bd-id* **Example:** `Device(config-vfi)# bridge-domain 100`	Specifies a bridge domain.
Step 7	**end** **Example:** `Device(config-vfi)# end`	Exits VFI configuration mode and returns to privileged EXEC mode.

Configuring a VFI on a PE Device: Alternate Configuration

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *name*
4. **vpn id** *id*
5. **member** *ip-address* [*vc-id*] **encapsulation mpls**
6. **exit**
7. **bridge-domain** *bd-id*
8. **member vfi** *vfi-name*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *name* **Example:** `Device(config)# l2vpn vfi context vfi110`	Establishes a L2VPN VFI between two or more separate networks, and enters VFI configuration mode.
Step 4	**vpn id** *id* **Example:**	Configures a VPN ID for a Virtual Private LAN Services (VPLS) domain. The emulated virtual circuits (VCs) bound

	Command or Action	Purpose
	`Device(config-vfi)# vpn id 110`	to this Layer 2 virtual routing and forwarding (VRF) instance use this VPN ID for signaling.
Step 5	**member** *ip-address* [*vc-id*] **encapsulation mpls** **Example:** `Device(config-vfi)# member 172.16.10.2 4 encapsulation mpls`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection and Multiprotocol Label Switching (MPLS) as the encapsulation type.
Step 6	**exit** **Example:** `Device(config-vfi)# exit`	Exits VFI configuration mode and returns to global configuration mode.
Step 7	**bridge-domain** *bd-id* **Example:** `Device(config)# bridge-domain 100`	Specifies a bridge domain and enters bridge-domain configuration mode.
Step 8	**member vfi** *vfi-name* **Example:** `Device(config-bdomain)# member vfi vfi110`	Binds a VFI instance to a bridge domain instance.
Step 9	**end** **Example:** `Device(config-bdomain)# end`	Exits bridge-domain configuration mode and returns to privileged EXEC mode.

Configuring Static Virtual Private LAN Services

To configure static Virtual Private LAN Services (VPLS), perform the following tasks:

- Configuring a Pseudowire for Static VPLS
- Configuring VFI for Static VPLS
- Configuring a VFI for Static VPLS: Alternate Configuration
- Configuring an Attachment Circuit for Static VPLS
- Configuring an Attachment Circuit for Static VPLS: Alternate Configuration
- Configuring an MPLS-TP Tunnel for Static VPLS with TP
- Configuring a VFI for Static VPLS: Alternate Configuration

Configuring a Pseudowire for Static VPLS

The configuration of pseudowires between provider edge (PE) devices helps in the successful transmission of the Layer 2 frames between PE devices.

Use the pseudowire template to configure the virtual circuit (VC) type for the virtual path identifier (VPI) pseudowire. In the following task, the pseudowire will go through a Multiprotocol Label Switching (MPLS)-Tunneling Protocol (TP) tunnel.

The pseudowire template configuration specifies the characteristics of the tunneling mechanism that is used by the pseudowires, which are:

- Encapsulation type
- Control protocol
- Payload-specific options
- Preferred path

Perform this task to configure a pseudowire template for static Virtual Private LAN Services (VPLS).

Note Ensure that you perform this task before configuring the virtual forwarding instance (VFI) peer. If the VFI peer is configured before the pseudowire class, the configuration is incomplete until the pseudowire class is configured. The **show running-config** command displays an error stating that configuration is incomplete.

```
Device# show running-config | sec vfi

l2 vfi config manual
 vpn id 1000
 ! Incomplete point-to-multipoint vfi config
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template type pseudowire** *name*
4. **encapsulation mpls**
5. **signaling protocol none**
6. **preferred-path interface Tunnel-tp** *interface-number*
7. **exit**
8. **interface pseudowire** *number*
9. **source template type pseudowire** *name*
10. **neighbor** *peer-address vcid-value*
11. **label** *local-pseudowire-label remote-pseudowire-label*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** *name* **Example:** `Device(config)# template type pseudowire static-vpls`	Specifies the template type as pseudowire and enters template configuration mode.
Step 4	**encapsulation mpls** **Example:** `Device(config-template)# encapsulation mpls`	Specifies the tunneling encapsulation. • For Any Transport over MPLS (AToM), the encapsulation type is MPLS.
Step 5	**signaling protocol none** **Example:** `Device(config-template)# signaling protocol none`	Specifies that no signaling protocol is configured for the pseudowire class.
Step 6	**preferred-path interface Tunnel-tp** *interface-number* **Example:** `Device(config-template)# preferred-path interface Tunnel-tp 1`	(Optional) Specifies the path that traffic uses: an MPLS Traffic Engineering (TE) tunnel or destination IP address and Domain Name Server (DNS) name.
Step 7	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to global configuration mode.
Step 8	**interface pseudowire** *number* **Example:** `Device(config)# interface pseudowire 1`	Establishes a pseudowire interface and enters interface configuration mode.
Step 9	**source template type pseudowire** *name* **Example:** `Device(config-if)# source template type pseudowire static-vpls`	Configures the source template type of the configured pseudowire.

	Command or Action	Purpose
Step 10	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.0.0.1 123`	Specifies the peer IP address and VC ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 11	**label** *local-pseudowire-label remote-pseudowire-label* **Example:** `Device(config-if)# label 301 17`	Configures an Any Transport over MPLS (AToM) static pseudowire connection by defining local and remote circuit labels.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring VFI for Static VPLS

Note Ensure that you perform this task after configuring the pseudowire. If the VFI peer is configured before the pseudowire, the configuration is incomplete until the pseudowire is configured. The output of the **show running-config** command displays an error stating that configuration is incomplete.

```
Device# show running-config | sec vfi

l2 vfi config manual
 vpn id 1000
 ! Incomplete point-to-multipoint vfi config
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label range** *minimum-value maximum-value* [**static** *minimum-static-value maximum-static-value*]
4. **pseudowire-class** [*pw-class-name*]
5. **encapsulation mpls**
6. **protocol** {**l2tpv2** | **l2tpv3** | **none**} *[l2tp-class-name]*
7. **exit**
8. **l2 vfi** *vfi-name* **manual**
9. **vpn id** *vpn-id*
10. **neighbor** *ip-address* **pw-class** *pw-name*
11. **mpls label** *local-pseudowire-label remote-pseudowire-label*
12. **mpls control-word**
13. **neighbor** *ip-address* **pw-class** *pw-name*
14. **mpls label** *local-pseudowire-label remote-pseudowire-label*
15. **mpls control-word**
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mpls label range** *minimum-value maximum-value* [**static** *minimum-static-value maximum-static-value*] **Example:** `Device(config)# mpls label range 16 200 static 300 500`	Configures the range of local labels available for use with Multiprotocol Label Switching (MPLS) applications on packet interfaces.
Step 4	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class static_vpls`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 5	**encapsulation mpls** **Example:** `Device(config-pw-class)# encapsulation mpls`	Specifies the tunneling encapsulation as MPLS.
Step 6	**protocol** {**l2tpv2** | **l2tpv3** | **none**} *[l2tp-class-name]* **Example:** `Device(config-pw-class)# protocol none`	Specifies that no signaling protocol will be used in Layer 2 Tunneling Protocol Version 3 (L2TPv3) sessions.
Step 7	**exit** **Example:** `Device(config-pw-class)# exit`	Exits pseudowire class configuration mode and returns to global configuration mode.
Step 8	**l2 vfi** *vfi-name* **manual** **Example:** `Device(config)# l2 vfi static-vfi manual`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) between two or more separate networks, and enters Layer 2 VFI manual configuration mode.
Step 9	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 100`	Specifies the VPN ID.

	Command or Action	Purpose
Step 10	**neighbor** *ip-address* **pw-class** *pw-name* **Example:** `Device(config-vfi)# neighbor 10.3.4.4 pw-class static_vpls`	Specifies the IP address of the peer and the pseudowire class.
Step 11	**mpls label** *local-pseudowire-label remote-pseudowire-label* **Example:** `Device(config-vfi)# mpls label 301 17`	Configures an Any Transport over MPLS (AToM) static pseudowire connection by defining local and remote circuit labels.
Step 12	**mpls control-word** **Example:** `Device(config-vfi)# mpls control-word`	(Optional) Enables the MPLS control word in an AToM static pseudowire connection.
Step 13	**neighbor** *ip-address* **pw-class** *pw-name* **Example:** `Device(config-vfi)# neighbor 2.3.4.3 pw-class static_vpls`	Specifies the IP address of the peer and the pseudowire class.
Step 14	**mpls label** *local-pseudowire-label remote-pseudowire-label* **Example:** `Device(config-vfi)# mpls label 302 18`	Configures an AToM static pseudowire connection by defining local and remote circuit labels.
Step 15	**mpls control-word** **Example:** `Device(config-vfi)# mpls control-word`	(Optional) Enables the MPLS control word in an AToM static pseudowire connection.
Step 16	**end** **Example:** `Device(config-vfi)# end`	Exits Layer 2 VFI manual configuration mode and returns to privileged EXEC mode.

Configuring a VFI for Static VPLS: Alternate Configuration

Note Ensure that you perform this task after configuring the pseudowire. If the VFI peer is configured before the pseudowire, the configuration is incomplete until the pseudowire is configured. The output of the **show running-config** command displays an error stating that configuration is incomplete.

```
Device# show running-config | sec vfi

l2 vfi config manual
 vpn id 1000
 ! Incomplete point-to-multipoint vfi config
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **exit**
6. **interface** *type number*
7. **encapsulation mpls**
8. **neighbor** *ip-address vc-id*
9. **label** *local-pseudowire-label remote-pseudowire-label*
10. **control-word** {**include** | **exclude**}
11. **exit**
12. **bridge-domain** *bd-id*
13. **member vfi** *vfi-name*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls1`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) between two or more separate networks and enters VFI configuration mode.

	Command or Action	Purpose
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 100`	Specifies the VPN ID.
Step 5	**exit** **Example:** `Device(config-vfi)# exit`	Exits VFI configuration mode and returns to global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface pseudowire 100`	Specifies an interface and enters interface configuration mode.
Step 7	**encapsulation mpls** **Example:** `Device(config-if)# encapsulation mpls`	Specifies an encapsulation type for tunneling Layer 2 traffic over a pseudowire.
Step 8	**neighbor** *ip-address vc-id* **Example:** `Device(config-if)# neighbor 10.3.4.4 100`	Specifies the peer IP address and virtual circuit (VC) ID value of a Layer 2 VPN (L2VPN) pseudowire.
Step 9	**label** *local-pseudowire-label remote-pseudowire-label* **Example:** `Device(config-if)# label 301 17`	Configures an Any Transport over MPLS (AToM) static pseudowire connection by defining local and remote circuit labels.
Step 10	**control-word** {**include** | **exclude**} **Example:** `Device(config-if)# control-word include`	(Optional) Enables the Multiprotocol Label Switching (MPLS) control word in an AToM dynamic pseudowire connection.
Step 11	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 12	**bridge-domain** *bd-id* **Example:** `Device(config)# bridge-domain 24`	Specifies the bridge domain ID and enters bridge-domain configuration mode.
Step 13	**member vfi** *vfi-name* **Example:**	Binds a service instance to a bridge domain instance.

	Command or Action	Purpose
	`Device(config-bdomain)# member vfi vpls1`	
Step 14	**end** **Example:** `Device(config-bdomain)# end`	Exits bridge-domain configuration mode and returns to privileged EXEC mode.

Configuring an Attachment Circuit for Static VPLS

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot/interface*
4. **service instance** *si-id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **rewrite ingress tag pop** *number* [**symmetric**]
7. **bridge-domain** *bd-id*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot/interface* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode. • Ensure that the interfaces between the customer edge (CE) and provider edge (PE) devices that run Ethernet over MPLS (EoMPLS) are in the same subnet. All other interfaces and backbone devices do not need to be in the same subnet.
Step 4	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 100 ethernet`	Configures an Ethernet service instance on an interface and enters service instance configuration mode.

	Command or Action	Purpose
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 200`	Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance. • Ensure that the interface on the adjoining CE device is on the same VLAN as this PE device.
Step 6	**rewrite ingress tag pop** *number* [**symmetric**] **Example:** `Device(config-if-srv)# rewrite ingress tag pop 1 symmetric`	(Optional) Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance and the tag to be removed from a packet.
Step 7	**bridge-domain** *bd-id* **Example:** `Device(config-if-srv)# bridge-domain 24`	(Optional) Binds a service instance or a MAC tunnel to a bridge domain instance.
Step 8	**end** **Example:** `Device(config-if-srv)# end`	Exits service instance configuration mode and returns to privileged EXEC mode.

Configuring an Attachment Circuit for Static VPLS: Alternate Configuration

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot/interface*
4. **service instance** *si-id* **ethernet**
5. **encapsulation dot1q** *vlan-id*
6. **rewrite ingress tag pop** *number* [**symmetric**]
7. **exit**
8. **exit**
9. **bridge-domain** *bd-id*
10. **member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*]
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *slot/interface* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode. • Ensure that the interfaces between the customer edge (CE) and provider edge (PE) devices that are running Ethernet over MPLS (EoMPLS) are in the same subnet. All other interfaces and backbone devices do not need to be in the same subnet.
Step 4	**service instance** *si-id* **ethernet** **Example:** `Device(config-if)# service instance 10 ethernet`	Specifies a service instance ID and enters service instance configuration mode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Device(config-if-srv)# encapsulation dot1q 200`	Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance. • Ensure that the interface on the adjoining CE device is on the same VLAN as this PE device.
Step 6	**rewrite ingress tag pop** *number* [**symmetric**] **Example:** `Device(config-if-srv)# rewrite ingress tag pop 1 symmetric`	(Optional) Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance and the tag to be removed from a packet.
Step 7	**exit** **Example:** `Device(config-if-srv)# exit`	Exits service instance configuration mode and returns to interface configuration mode.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**bridge-domain** *bd-id* **Example:** `Device(config)# bridge-domain 100`	Specifies the bridge domain ID and enters bridge-domain configuration mode.

	Command or Action	Purpose
Step 10	**member** *interface-type-number* **service-instance** *service-id* [**split-horizon group** *group-id*] **Example:** `Device(config-bdomain)# member gigabitethernet0/0/1 service-instance 1000`	(Optional) Binds a service instance to a bridge domain instance.
Step 11	**end** **Example:** `Device(config-bdomain)# end`	Exits bridge-domain configuration mode and returns to privileged EXEC mode.

Configuring an MPLS-TP Tunnel for Static VPLS with TP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface Tunnel-tp** *number*
4. **no ip address**
5. **no keepalive**
6. **tp destination** *ip-address*
7. **bfd** *bfd-template*
8. **working-lsp**
9. **out-label** *number* **out-link** *number*
10. **lsp-number** *number*
11. **exit**
12. **protect-lsp**
13. **out-label** *number* **out-link** *number*
14. **in-label** *number*
15. **lsp-number** *number*
16. **exit**
17. **exit**
18. **interface** *type number*
19. **ip address** *ip-address ip-mask*
20. **mpls tp link** *link-num* {**ipv4** *ip-address* | **tx-mac** *mac-address*}
21. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface Tunnel-tp** *number* **Example:** `Device(config)# interface Tunnel-tp 4`	Configures a Multiprotocol Label Switching (MPLS) transport profile tunnel and enters interface configuration mode. • Use the same interface as you configured for the pseudowire class.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables the IP address configuration.
Step 5	**no keepalive** **Example:** `Device(config-if)# no keepalive`	Disables the keepalive configuration.
Step 6	**tp destination** *ip-address* **Example:** `Device(config-if)# tp destination 10.22.22.22`	Configures the tunnel destination.
Step 7	**bfd** *bfd-template* **Example:** `Device(config-if)# bfd tp`	Binds a single-hop Bidirectional Forwarding Detection (BFD) template to an interface.
Step 8	**working-lsp** **Example:** `Device(config-if)# working-lsp`	Configures the working label switched path (LSP) and enters working interface configuration mode.
Step 9	**out-label** *number* **out-link** *number* **Example:** `Device(config-if-working)# out-label 16 out-link 100`	Configures the out link and out label for the working LSP.
Step 10	**lsp-number** *number* **Example:**	Configures the ID number for the working LSP.

	Command or Action	Purpose
	`Device(config-if-working)# lsp-number 0`	
Step 11	**exit** **Example:** `Device(config-if-working)# exit`	Exits working interface configuration mode and returns to interface configuration mode.
Step 12	**protect-lsp** **Example:** `Device(config-if)# protect-lsp`	Enters protection configuration mode for the label switched path (LSP) and enters protect interface configuration mode.
Step 13	**out-label** *number* **out-link** *number* **Example:** `Device(config-if-protect)# out-label 11 out-link 500`	Configures the out link and out label for the protect LSP.
Step 14	**in-label** *number* **Example:** `Device(config-if-protect)# in-label 600`	Configures the in label for the protect LSP.
Step 15	**lsp-number** *number* **Example:** `Device(config-if-protect)# lsp-number 1`	Configures the ID number for the working protect LSP.
Step 16	**exit** **Example:** `Device(config-if-protect)# exit`	Exits protect interface configuration mode and returns to interface configuration mode.
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 18	**interface** *type number* **Example:** `Device(config-if)# interface GigabitEthernet 0/1/0`	Configures a interface and enters interface configuration mode.
Step 19	**ip address** *ip-address ip-mask* **Example:** `Device(config)# ip address 10.0.0.1 255.255.255.0`	(Optional) Configures the IP address and mask if not using an IP-less core.

	Command or Action	Purpose
Step 20	**mpls tp link** *link-num* {**ipv4** *ip-address* | **tx-mac** *mac-address*} **Example:** `Device(config-if)# mpls tp link 10 tx-mac 0100.0c99.8877`	Configures Multiprotocol Label Switching (MPLS) transport profile (TP) link parameters.
Step 21	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for Virtual Private LAN Services

Example: Configuring 802.1Q Access Ports for Tagged Traffic from a CE Device

This example shows how to configure the tagged traffic:

```
Device(config)# interface GigabitEthernet 0/0/1
Device(config-if)# no ip address
Device(config-if)# negotiation auto
Device(config-if)# service instance 10 ethernet
Device(config-if-srv)# encapsulation dot1q 200
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# end
```

Example: Configuring 802.1Q Access Ports for Tagged Traffic from a CE Device: Alternate Configuration

The following example shows how to configure the tagged traffic:

```
Device(config)# interface GigabitEthernet 0/0/1
Device(config-if)# no ip address
Device(config-if)# negotiation auto
Device(config-if)# service instance 10 ethernet
Device(config-if-srv)# encapsulation dot1q 200
Device(config-if-srv)# exit
Device(config-if)# exit
Device(config)# bridge-domain 100
Device(config-bdomain)# member gigabitethernet0/0/1 service-instance 1000
Device(config-bdomain)# end
```

Example: Configuring Access Ports for Untagged Traffic from a CE Device

The following example shows how to configure access ports for untagged traffic:

```
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# no ip address
Device(config-if)# negotiation auto
Device(config-if)# service instance 10 ethernet
Device(config-if-srv)# encapsulation untagged
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# end
```

The following example shows a virtual forwarding interface (VFI) configuration:

```
Device(config)# l2 vfi VPLSA manual
Device(config-vfi)# vpn id 110
Device(config-vfi)# neighbor 10.11.11.11 encapsulation mpls
Device(config-vfi)# neighbor 10.33.33.33 encapsulation mpls
Device(config-vfi)# neighbor 10.44.44.44 encapsulation mpls
Device(config-vfi)# bridge-domain 110
Device(config-vfi)# end
```

The following example shows a VFI configuration for hub and spoke.

```
Device(config)# l2 vfi VPLSB manual
Device(config-vfi)# vpn id 111
Device(config-vfi)# neighbor 10.99.99.99 encapsulation mpls
Device(config-vfi)# neighbor 10.12.12.12 encapsulation mpls
Device(config-vfi)# neighbor 10.13.13.13 encapsulation mpls no-split-horizon
Device(config-vfi)# bridge-domain 111
Device(config-vfi)# end
```

The output of the **show mpls 12transport vc** command displays various information related to a provide edge (PE) device. The VC ID in the output represents the VPN ID; the VC is identified by the combination of the destination address and the VC ID as shown in the command output. The output of the **show mpls l2transport vc detail** command displays detailed information about virtual circuits (VCs) on a PE device.

```
Device# show mpls l2transport vc 201

Local intf     Local circuit        Dest address    VC ID      Status
-------------  -------------------- --------------- ---------- ----------
VFI VPLSA      VFI                  10.11.11.11     110        UP
VFI VPLSA      VFI                  10.33.33.33     110        UP
VFI VPLSA      VFI                  10.44.44.44     110        UP
```

The following sample output from the **show vfi** command displays the VFI status:

```
Device# show vfi VPLSA

VFI name: VPLSA, state: up
  Local attachment circuits:
    Vlan2
  Neighbors connected via pseudowires:
  Peer Address     VC ID     Split-horizon
  10.11.11.11          110            Y
  10.33.33.33          110            Y
  10.44.44.44          110            Y


Device# show vfi VPLSB
```

```
VFI name: VPLSB, state: up
  Local attachment circuits:
    Vlan2
  Neighbors connected via pseudowires:
  Peer Address     VC ID     Split-horizon
  10.99.99.99       111            Y
  10.12.12.12       111            Y
  10.13.13.13       111            N
```

Example: Configuring Access Ports for Untagged Traffic from a CE Device: Alternate Configuration

The following example shows how to configure the untagged traffic.

```
Device(config)# interface GigabitEthernet 0/4/4
Device(config-if)# no ip address
Device(config-if)# negotiation auto
Device(config-if)# service instance 10 ethernet
Device(config-if-srv)# encapsulation untagged
Device(config-if-srv)# exit
Device(config-if)# exit
Device(config)# bridge-domain 100
Device(config-bdomain)# member GigabitEthernet0/4/4 service-instance 10
Device(config-if-srv)# end
```

Example: Configuring Q-in-Q EFP

The following example shows how to configure the tagged traffic.

```
Device(config)# interface GigabitEthernet 0/0/2
Device(config-if)# no ip address
Device(config-if)# negotiate auto
Device(config-if)# service instance 10 ethernet
Device(config-if-srv)# encapsulation dot1q 200 second-dot1q 400
Device(config-if-srv)# bridge-domain 100
Device(config-if-srv)# end
```

Use the **show spanning-tree vlan** command to verify that the ports are not in a blocked state. Use the **show vlan id** command to verify that a specific port is configured to send and receive specific VLAN traffic.

Example: Configuring Q-in-Q in EFP: Alternate Configuration

The following example shows how to configure the tagged traffic:

```
Device(config)# interface GigabitEthernet 0/4/4
Device(config-if)# no ip address
Device(config-if)# nonegotiate auto
Device(config-if)# service instance 10 ethernet
Device(config-if-srv)# encapsulation dot1q 200 second-dot1q 400
Device(config-if-srv)# exit
Device(config-if)# exit
```

```
Device(config)# bridge-domain 100
Device(config-bdomain)# member GigabitEthernet0/4/4 service-instance 1000
Device(config-bdomain)# end
```

Use the **show spanning-tree vlan** command to verify that the port is not in a blocked state. Use the **show vlan id** command to verify that a specific port is configured to send and receive a specific VLAN traffic.

Example: Configuring MPLS on a PE Device

The following example shows a global Multiprotocol Label Switching (MPLS) configuration:

```
Device(config)# mpls label protocol ldp
Device(config)# mpls ldp logging neighbor-changes
Device(config)# mpls ldp discovery hello holdtime 5
Device(config)# mpls ldp router-id Loopback0 force
```

The following sample output from the **show ip cef** command displays the Label Distribution Protocol (LDP) label assigned:

```
Device# show ip cef 192.168.17.7

192.168.17.7/32, version 272, epoch 0, cached adjacency to POS4/1
0 packets, 0 bytes
  tag information set
    local tag: 8149
    fast tag rewrite with PO4/1, point2point, tags imposed: {4017}
  via 10.3.1.4, POS4/1, 283 dependencies
    next hop 10.3.1.4, POS4/1
    valid cached adjacency
    tag rewrite with PO4/1, point2point, tags imposed: {4017}
```

Example: VFI on a PE Device

The following example shows a virtual forwarding instance (VFI) configuration:

```
Device(config)# l2 vfi vfi110 manual
Device(config-vfi)# vpn id 110
Device(config-vfi)# neighbor 172.16.10.2 4 encapsulation mpls
Device(config-vfi)# neighbor 10.16.33.33 encapsulation mpls
Device(config-vfi)# neighbor 198.51.100.44 encapsulation mpls
Device(config-vfi)# bridge-domain 100
Device(config-vfi)# end
```

The following example shows a VFI configuration for a hub-and-spoke configuration:

```
Device(config)# l2 vfi VPLSA manual
Device(config-vfi)# vpn id 110
Device(config-vfi)# neighbor 10.9.9.9 encapsulation mpls
Device(config-vfi)# neighbor 192.0.2.12 encapsulation mpls
Device(config-vfi)# neighbor 203.0.113.4 encapsulation mpls no-split-horizon
Device(config-vfi)# bridge-domain 100
Device(config-vfi)# end
```

The **show mpls 12transport vc** command displays information about the provider edge (PE) device. The **show mpls l2transport vc detail** command displays detailed information about the virtual circuits (VCs) on a PE device.

```
Device# show mpls l2transport vc 201

Local intf     Local circuit        Dest address    VC ID      Status
-------------  -------------------- --------------- ---------- ----------
VFI test1      VFI                  209.165.201.1   201        UP
VFI test1      VFI                  209.165.201.2   201        UP
VFI test1      VFI                  209.165.201.3   201        UP
```

The **show vfi** *vfi-name* command displays VFI status. The VC ID in the output represents the VPN ID; the VC is identified by the combination of the destination address and the VC ID as in the example below.

```
Device# show vfi VPLS-2

VFI name: VPLS-2, state: up
  Local attachment circuits:
    Vlan2
  Neighbors connected via pseudowires:
  Peer Address     VC ID     Split-horizon
  10.1.1.1          2             Y
  10.1.1.2          2             Y
  10.2.2.3          2             N
```

Example: VFI on a PE Device: Alternate Configuration

The following example shows how to configure a virtual forwarding interface (VFI) on a provider edge (PE) device:

```
Device(config)# l2vpn vfi context vfi110
Device(config-vfi)# vpn id 110
Device(config-vfi)# member 172.16.10.2 4 encapsulation mpls
Device(config-vfi)# member 10.33.33.33 encapsulation mpls
Device(config-vfi)# member 10.44.44.44 encapsulation mpls
Device(config-vfi)# exit
Device(config)# bridge-domain 100
Device(config-bdomain)# member vfi vfi110
Device(config-bdomain)# end
```

The following example shows how to configure a hub-and-spoke VFI configuration:.

```
Device(config)# l2vpn vfi context VPLSA
Device(config-vfi)# vpn id 110
Device(config-vfi)#  member 10.9.9.9 encapsulation mpls
Device(config-vfi)#  member 172.16.10.2 4 encapsulation mpls
Device(config-vfi)#  exit
Device(config)# bridge-domain 100
Device(config-bdomain)# member vfi VPLSA
Device(config-bdomain)# member GigabitEthernet0/0/0 service-instance 100
Device(config-bdomain)# member 10.33.33.33 10 encapsulation mpls
Device(config-bdomain)# end
```

The **show l2vpn atom vc** command displays information about the PE device. The command also displays information about Any Transport over MPLS (AToM) virtual circuits (VCs) and static pseudowires that are enabled to route Layer 2 packets on a device.

```
Device# show l2vpn atom vc

Local intf     Local circuit            Dest address    VC ID      Status
------------- ----------------------- --------------- ---------- ----------
Et0/0.1        Eth VLAN 101             10.0.0.2        101        UP
Et0/0.1        Eth VLAN 101             10.0.0.3        201        DOWN
```

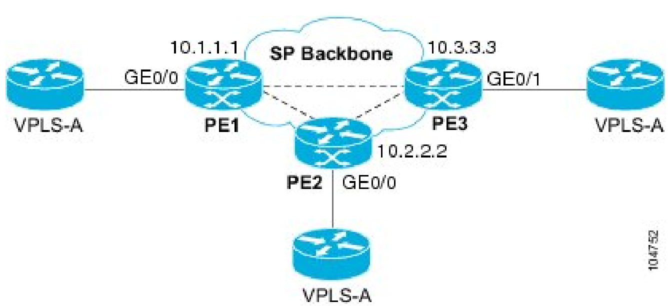
The **show l2vpn vfi** command displays the VFI status. The VC ID in the output represents the VPN ID; the VC is identified by the combination of the destination address and the VC ID as in the example below.

```
Device# show l2vpn vfi VPLS-2

Legend: RT= Route-target

VFI name: serviceCore1, State: UP, Signaling Protocol: LDP
  VPN ID: 100, VPLS-ID: 9:10, Bridge-domain vlan: 100
  RD: 9:10, RT: 10.10.10.10:150
  Pseudo-port Interface: Virtual-Ethernet1000

  Neighbors connected via pseudowires:
  Interface     Peer Address    VC ID      Discovered Router ID   Next Hop
  Pw2000        10.0.0.1        10         10.0.0.1               10.0.0.1
  Pw2001        10.0.0.2        10         10.1.1.2               10.0.0.2
  Pw2002        10.0.0.3        10         10.1.1.3               10.0.0.3
  Pw5           10.0.0.4        10         -                      10.0.0.4
```

Example: Full-Mesh VPLS Configuration

In a full-mesh configuration, each provider edge (PE) device creates a multipoint-to-multipoint forwarding relationship with all other PE devices in the Virtual Private LAN Services (VPLS) domain using a virtual forwarding interface (VFI). An Ethernet or a VLAN packet received from the customer network can be forwarded to one or more local interfaces and/or emulated virtual circuits (VCs) in the VPLS domain. To avoid a broadcast packet loop in the network, packets received from an emulated VC cannot be forwarded to any emulated VC in the VPLS domain on a PE device. Ensure that Layer 2 split horizon is enabled to avoid a broadcast packet loop in a full-mesh network.

Figure 42: Full-Mesh VPLS Configuration



PE 1 Configuration

The following examples shows how to create virtual switch instances (VSIs) and associated VCs:

```
l2 vfi PE1-VPLS-A manual
 vpn id 100
 neighbor 10.2.2.2 encapsulation mpls
 neighbor 10.3.3.3 encapsulation mpls
 bridge domain 100
!
interface Loopback 0
 ip address 10.1.1.1 255.255.0.0
```

The following example shows how to configure the customer edge (CE) device interface (there can be multiple Layer 2 interfaces in a VLAN):

```
interface GigabitEthernet 0/0/0
 no ip address
 negotiation auto
 service instance 10 ethernet
 encapsulation dot1q 200
 bridge-domain 100
```

PE 2 Configuration

The following example shows how to create VSIs and associated VCs.

```
l2 vfi PE2-VPLS-A manual
 vpn id 100
 neighbor 10.1.1.1 encapsulation mpls
 neighbor 10.3.3.3 encapsulation mpls
 bridge domain 100
!
interface Loopback 0
 ip address 10.2.2.2 255.255.0.0
```

The following example shows how to configure the CE device interface (there can be multiple Layer 2 interfaces in a VLAN):

```
interface GigabitEthernet 0/0/0
 no ip address
 negotiation auto
 service instance 10 ethernet
 encapsulation dot1q 200
 bridge-domain 100
```

PE 3 Configuration

The following example shows how to create VSIs and associated VCs:

```
l2 vfi PE3-VPLS-A manual
 vpn id 112
 neighbor 10.1.1.1 encapsulation mpls
 neighbor 10.2.2.2 encapsulation mpls
 bridge domain 100
!
interface Loopback 0
 ip address 10.3.3.3 255.255.0.0
```

The following example shows how to configure the CE device interface (there can be multiple Layer 2 interfaces in a VLAN).

```
interface GigabitEthernet 0/0/1
 no ip address
 negotiation auto
 service instance 10 ethernet
 encapsulation dot1q 200
 bridge-domain 100
!
```

The following sample output from the **show mpls l2 vc** command provides information about the status of the VC:

```
Device# show mpls l2 vc

Local intf       Local circuit        Dest address    VC ID      Status
-------------    -------------------- --------------- ---------- ----------
VFI PE1-VPLS-A   VFI                  10.2.2.2         100        UP
VFI PE1-VPLS-A   VFI                  10.3.3.3         100        UP
```

The following sample output from the **show vfi** command provides information about the VFI:

```
Device# show vfi PE1-VPLS-A

VFI name: VPLSA, state: up
  Local attachment circuits:
    Vlan200
  Neighbors connected via pseudowires:
    10.2.2.2  10.3.3.3
```

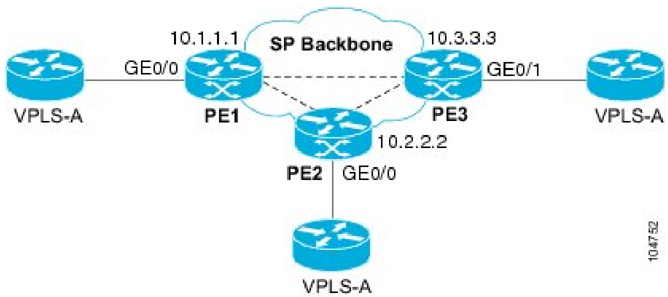
The following sample output from the **show mpls 12transport vc** command provides information about virtual circuits:

```
Device# show mpls l2transport vc detail

Local interface: VFI PE1-VPLS-A up
  Destination address: 10.2.2.2, VC ID: 100, VC status: up
    Tunnel label: imp-null, next hop point2point
    Output interface: Se2/0, imposed label stack {18}
  Create time: 3d15h, last status change time: 1d03h
  Signaling protocol: LDP, peer 10.2.2.2:0 up
    MPLS VC labels: local 18, remote 18
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 0, send 0
    byte totals:   receive 0, send 0
    packet drops:  receive 0, send 0
```

Example: Full-Mesh Configuration : Alternate Configuration

In a full-mesh configuration, each provider edge (PE) router creates a multipoint-to-multipoint forwarding relationship with all other PE routers in the Virtual Private LAN Services (VPLS) domain using a virtual forwarding interface (VFI). An Ethernet or virtual LAN (VLAN) packet received from the customer network can be forwarded to one or more local interfaces and/or emulated virtual circuits (VCs) in the VPLS domain. To avoid broadcasted packets looping in the network, no packet received from an emulated VC can be forwarded to any emulated VC of the VPLS domain on a PE router. That is, Layer 2 split horizon should always be enabled as the default in a full-mesh network.

Figure 43: VPLS Configuration Example



PE 1 Configuration

The following example shows how to create virtual switch instances (VSIs) and associated VCs and to configure the CE device interface (there can be multiple Layer 2 interfaces in a VLAN):

```
interface gigabitethernet 0/0/0
 service instance 100 ethernet
 encap dot1q 100
 no shutdown
!
l2vpn vfi context PE1-VPLS-A
 vpn id 100
 neighbor 10.2.2.2 encapsulation mpls
 neighbor 10.3.3.3 encapsulation mpls
!
bridge-domain 100
 member gigabitethernet0/0/0 service-instance 100
 member vfi PE1-VPLS-A
```

PE 2 Configuration

The following example shows how to create VSIs and associated VCs and to configure the CE device interface (there can be multiple Layer 2 interfaces in a VLAN):

```
interface gigabitethernet 0/0/0
 service instance 100 ethernet
 encap dot1q 100
 no shutdown
!
l2vpn vfi context PE2-VPLS-A
 vpn id 100
 neighbor 10.1.1.1 encapsulation mpls
 neighbor 10.3.3.3 encapsulation mpls
!
bridge-domain 100
 member gigabitethernet0/0/0 service-instance 100
 member vfi PE2-VPLS-A
```

PE 3 Configuration

The following example shows how to create of the VSIs and associated VCs and to configure the CE device interface (there can be multiple Layer 2 interfaces in a VLAN):

```
interface gigabitethernet 0/0/0
 service instance 100 ethernet
 encap dot1q 100
 no shutdown
!
l2vpn vfi context PE3-VPLS-A
 vpn id 100
 neighbor 10.1.1.1 encapsulation mpls
 neighbor 10.2.2.2 encapsulation mpls
!
bridge-domain 100
 member gigabitethernet0/0/0 service-instance 100
 member vfi PE3-VPLS-A
```

The following sample output from the **show mpls l2 vc** command provides information on the status of the VC:

```
Device# show mpls l2 vc

Local intf       Local circuit  Dest address    VC ID      Status
-------------    -------------- --------------- ---------- ----------
VFI PE3-VPLS-A   VFI            10.2.2.2        100        UP
VFI PE3-VPLS-A   VFI            10.3.3.3        100        UP
```

The following sample output from the **show l2vpn vfi** command provides information about the VFI:

```
Device# show l2vpn vfi VPLS-2

Legend: RT= Route-target

VFI name: serviceCore1, State: UP, Signaling Protocol: LDP
  VPN ID: 100, VPLS-ID: 9:10, Bridge-domain vlan: 100
  RD: 9:10, RT: 10.10.10.10:150
  Pseudo-port Interface: Virtual-Ethernet1000

  Neighbors connected via pseudowires:
  Interface    Peer Address    VC ID      Discovered Router ID   Next Hop
  Pw2000       10.0.0.1        10         10.0.0.1               10.0.0.1
  Pw2001       10.0.0.2        10         10.1.1.2               10.0.0.2
  Pw2002       10.0.0.3        10         10.1.1.3               10.0.0.3
  Pw5          10.0.0.4        10         -                      10.0.0.4
```

The following sample output from the **show l2vpn atom vc** command provides information on the virtual circuits:

```
Device# show l2vpn atom vc

Local intf    Local circuit           Dest address    VC ID      Status
------------- ----------------------- --------------- ---------- ----------
Et0/0.1       Eth VLAN 101            10.0.0.2        101        UP
Et0/0.1       Eth VLAN 101            10.0.0.3        201        DOWN
```

Example: MAC ACL with Dummy VLAN ID

PE basic configuration for VPLS type 4

```
 router bgp 100
 bgp log-neighbor-changes
 neighbor 19.0.0.1 remote-as 100
 neighbor 19.0.0.1 update-source Loopback0
 !
 address-family ipv4
  neighbor 19.0.0.1 activate
  neighbor 19.0.0.1 send-community extended
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 19.0.0.1 activate
 exit-address-family
l2vpn vfi context vlan_tag
 vpn id 10
 autodiscovery bgp signaling ldp template vlan_tag
!
mpls label protocol ldp
bridge-domain 10
 member GigabitEthernet2/1/0 service-instance 10
  remote circuit id 191
 member vfi vlan_tag
template type pseudowire vlan_tag
 encapsulation mpls
 vc type vlan
 control-word include
interface GigabitEthernet2/1/0
 no ip address
 negotiation auto
 service instance 10 ethernet
  encapsulation dot1q 10
 !
interface GigabitEthernet2/1/4
 ip address 108.0.0.2 255.255.255.0
 negotiation auto
 mpls ip
!

//Change the circuit ID and check if the download ID is correct//
bridge-domain 10
 member gigabitEthernet 2/1/0 service-instance 10
  remote circuit id 1982 <<< Set the dummy VLAN
```

Verifying the Configuration

Here's a sample output for the **show** command to verify the configured VLAN ID.

```
Device# show platform hardware qfp active feature bridge-domain client 10 interface

QFP L2BD datapath interface information
Name: GigabitEthernet2/1/0.EFP10
IF handle: 26, Input uidb: 245752
Flags: 0X000038
Split-horizon cfged: No, shg id: 0
STP state: Unknown/Bad
Mac security enabled:
```

```
MAC limit: 65536, MAC learned: 0
BD PPE addr: 0X8CBF3C00
efp circuit id: 1982 <<< The configured VLAN ID
```

Feature Information for Configuring Virtual Private LAN Services

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 32: Feature Information for Configuring Virtual Private LAN Services

Feature Name	Releases	Feature Information
Virtual Private LAN Services (VPLS)	Cisco IOS XE Release 3.5S	This feature enables you to configure dynamic Virtual Private LAN Services (VPLS). VPLS is a class of VPN that supports the connection of multiple sites in a single bridged domain over a managed IP/MPLS network. In Cisco IOS XE Release 3.5S, this feature was introduced on the Cisco ASR 903 Series Aggregation Services Routers.
L2VPN Protocol-Based CLIs	Cisco IOS XE Release 3.7S	In Cisco IOS XE Release 3.7S, the L2VPN Protocol-Based CLIs feature was introduced. This feature provides a set of processes and an improved infrastructure for developing and delivering Cisco IOS software on various Cisco platforms. This feature introduces new commands and modifies or replaces existing commands to achieve a consistent functionality across Cisco platforms and provide cross-Operating System support.
Static VPLS over MPLS-TP	Cisco IOS XE Release 3.6S	This features enables static VPLS to use MPLS Transport Profile. In Cisco IOS XE Release 3.6S, this feature was introduced on the Cisco ASR 903 Series Aggregation Services Routers.
Type 4 PWE VLAN Rewrite	Cisco IOS XE Everest Release 16.4.1	From Cisco IOS XE Everest 16.4.1 release, VPLS VC type 4 mode (with autodiscovery) can be used to configure a dummy VLAN tag. This feature can be used to modify the VLAN ID to filter based on the VLAN ID.

CHAPTER 21

Routed Pseudo-Wire and Routed VPLS

This feature module explains how to configure Routed Pseudo-Wire and Routed VPLS .

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Configuring Routed Pseudo-Wire and Routed VPLS

RPW and Routed VPLS can route Layer 3 traffic as well as switch Layer 2 frames for pseudowire connections between provider edge (PE) devices. Both point-to-point PE connections, in the form of Ethernet over MPLS (EoMPLS), and Virtual Private LAN Services (VPLS) multipoint PE connections are supported. The ability to route frames to and from these interfaces supports termination of a pseudowire into a Layer 3 network (VPN or global) on the same switch, or to tunnel Layer 3 frames over a Layer 2 tunnel (EoMPLS or VPLS). The feature supports faster network convergence in the event of a physical interface or device failure through the MPLS Traffic Engineering (MPLS-TE) and Fast Reroute (FRR) features. In particular, the feature enables MPLS TE-FRR protection for Layer 3 multicast over a VPLS domain.

When the RPW is configured in A-VPLS mode, TE/FRR is not supported because A-VPLS runs over ECMP and the ECMP convergence is comparable to TE/FRR.

To configure routing support for the pseudowire, configure an IP address and other Layer 3 features for the Layer 3 domain (VPN or global) in the virtual LAN (VLAN) interface configuration. The following example assigns the IP address 10.10.10.1 to the VLAN 100 interface, and enables Multicast PIM. (Layer 2 forwarding is defined by the VFI VFI100.)

```
interface bdi 100


  ip address 10.10.10.1 255.255.255.0
```

The following example assigns an IP address 20.20.20.1 of the VPN domain VFI200. (Layer 2 forwarding is defined by the VFI VFI200.)

```
interface bdi 200


  ip address 20.20.20.1 255.255.255.0
```

Verifying Routed Pseudo-Wire and Routed VPLS Configuration

You can use the **show mpls platform** command to view information about a routed pseudowire and routed VPLS configuration.

The following example shows how to display information about a routed pseudowire and routed VPLS configuration:

SUMMARY STEPS

1. show mpls platform vpls 100

DETAILED STEPS

show mpls platform vpls 100

Example:

```
Device# show mpls platform vpls 100

----------------------------
VPLS VLAN 100 (BD 100): V4
  VC info (#spoke VCs 0) :
    Imp:  tcam 224    (68    ) adj 131076 (0x20004) [peer 1.1.1.1 ID vc_id 100 2:1]  \
stats 0/0 0/0
    Disp: tcam 324    (66    ) adj 114692 (0x1C004) [in_label 16] stats 0/0
-------------------------------
BD Flood Manager: VLAN/BD 100, 3 peers, V4
  CMET handle 0x8 top 8 (0x8) bottom 3280 (0xCD0)
  Ingr flood: tcam 64/0x40 (sw 15) adj 196608   (0x30000) elif 0x701C0064 stats 0/0  \
0/0
  Egr  flood: tcam 65/0x41 (sw 72) adj 180228   (0x2C004) elif 0x701C0064 stats 0/0  \
0/0
    BD ports:       adj 32868    (0x8064) elif 0x20000064 stats 3/208
  Ingr local: tcam 32/0x20 (sw 13) adj 180224   (0x2C000) elif 0x20000064 stats 0/0
  Egr  local: tcam 33/0x21 (sw 14) adj 180225   (0x2C001) elif 0x20000064 stats 0/0
  IRB Ingr V4 Mcast control 162/0xA2 (sw 79), adj 196609   (0x30001)
      Egr  V4 Mcast control 164/0xA4 (sw 84), adj 180229   (0x2C005)
      Ingr V4 Mcast data 192/0xC0 (sw 80), adj 1966
(0x30000)
      Egr  V4 Mcast data 194/0xC2 (sw 85), adj 180228   (0x2C004)
```

```
      Ingr V4 Bcast 34/0x22 (sw 81), adj 196609   (0x30001)
      Egr  V4 Bcast 35/0x23 (sw 86), adj 180229   (0x2C005)
  IRB Ingr V6 Mcast control 608/0x260 (sw 82), adj 196608   (0x30000)
      Egr  V6 Mcast control 612/0x264 (sw 89), adj 180228   (0x2C004)
      Ingr V6 Mcast data 672/0x2A0 (sw 83), adj 196608   (0x30000)
      Egr  V6 Mcast data 676/0x2A4 (sw 90), adj 180228   (0x2C004)
      ip2irb local 36/0x24 (sw 87), adj 180226   (0x2C002) stats 0/0
      ip2irb flood 66/0x42 (sw 88), adj 180230   (0x2C006) stats 0/0
BD Flood Manager: 1 BDs, LTL base 0x90E, LTL clients: VPLS
                : Wildcard entry tcam 288 (12) adj 78089 (0x13109)
```

Feature Information for Routed Pseudo-Wire and Routed VPLS

Table 33: Feature Information for Routed Pseudo-Wire and Routed VPLS

Feature Name	Releases	Feature Information
Routed Pseudo-Wire and Routed VPLS	12.2(33)SRB 12.2(33)SXJ1 15.0(1)SY 15.2(4)M Cisco IOS XE Release 3.6S	This feature routes Layer 3 traffic as well as switch Layer 2 frames for pseudowire connections between provider edge (PE) devices. In Cisco IOS Release 12.2(33)SRB, this feature was introduced on the Cisco 7600 series routers. In Cisco IOS Release 12.2(33)SXJ1, this feature was integrated. This feature is supported on WAN cards. The following command was modified: **show mpls platform** In Cisco IOS Release 15.0(1)SY, this feature was integrated. In Cisco IOS Release 15.2(4)M, this feature was integrated. In Cisco IOS XE Release 3.6S, support was added for the Cisco ASR 1000 Series Routers.

CHAPTER 22

VPLS Autodiscovery BGP Based

VPLS Autodiscovery enables Virtual Private LAN Service (VPLS) provider edge (PE) devices to discover other PE devices that are part of the same VPLS domain. VPLS Autodiscovery also automatically detects when PE devices are added to or removed from a VPLS domain. As a result, with VPLS Autodiscovery enabled, you no longer need to manually configure a VPLS domain and maintain the configuration when a PE device is added or deleted. VPLS Autodiscovery uses the Border Gateway Protocol (BGP) to discover VPLS members and set up and tear down pseudowires in a VPLS domain.

This module describes how to configure BGP-based VPLS Autodiscovery.

Restrictions for VPLS Autodiscovery BGP Based

- Virtual Private LAN Service (VPLS) Autodiscovery supports only IPv4 addresses.
- VPLS Autodiscovery uses Forwarding Equivalence Class (FEC) 129 to convey endpoint information. Manually configured pseudowires use FEC 128.
- VPLS Autodiscovery is not supported with Layer 2 Tunnel Protocol Version 3 (L2TPv3).
- You can configure both autodiscovered and manually configured pseudowires in a single virtual forwarding instance (VFI). However, you cannot configure different pseudowires on the same peer PE device.
- After enabling VPLS Autodiscovery, if you manually configure a neighbor by using the **neighbor** command and both peers are in autodiscovery mode, each peer will receive discovery data for that VPLS. To prevent peers from receiving data for the VPLS domain, manually configure route target (RT) values.
- If you manually configure multiple pseudowires and target different IP addresses on the same PE device for each pseudowire, do not use the same virtual circuit (VC) ID to identify pseudowires that terminate at the same PE device.
- If you manually configure a neighbor on one PE device, you cannot configure the same pseudowire in the other direction by using autodiscovery on another PE device.

- Tunnel selection is not supported with autodiscovered neighbors.
- Up to 16 RTs are supported per VFI.
- The same RT is not allowed in multiple VFIs on the same PE device.
- The Border Gateway Protocol (BGP) autodiscovery process does not support dynamic, hierarchical VPLS. User-facing PE (U-PE) devices cannot discover network-facing PE (N-PE) devices, and N-PE devices cannot discover U-PE devices.
- Pseudowires for autodiscovered neighbors have split horizon enabled. (A split horizon is enabled by default on all interfaces. A split horizon blocks route information from being advertised by a device, irrespective of the interface from which the information originates.) Therefore, manually configure pseudowires for hierarchical VPLS. Ensure that U-PE devices do not participate in BGP autodiscovery for these pseudowires.
- Do not disable split horizon on autodiscovered neighbors. Split horizon is required with VPLS Autodiscovery.
- The provisioned peer address must be a /32 address bound to the peer's Label Distribution Protocol (LDP) router ID.
- A peer PE device must be able to access the IP address that is used as the local LDP router ID. Even if the IP address is not used in the **xconnect** command on the peer PE device, the IP address must be reachable.

Information About VPLS Autodiscovery BGP Based

How VPLS Works

Virtual Private LAN Service (VPLS) allows Multiprotocol Label Switching (MPLS) networks to provide multipoint Ethernet LAN services, also known as Transparent LAN Services (TLS). All customer sites in a VPLS appear to be on the same LAN, even though these sites might be in different geographic locations.

How the VPLS Autodiscovery BGP Based Feature Works

VPLS Autodiscovery enables each Virtual Private LAN Service (VPLS) provider edge (PE) device to discover other PE devices that are part of the same VPLS domain. VPLS Autodiscovery also tracks PE devices when they are added to or removed from a VPLS domain. Autodiscovery and signaling functions use the Border Gateway Protocol (BGP) to find and track PE devices.

BGP uses the Layer 2 VPN (L2VPN) Routing Information Base (RIB) to store endpoint provisioning information, which is updated each time any Layer 2 virtual forwarding instance (VFI) is configured. The prefix and path information is stored in the L2VPN database, which allows BGP to make decisions about the best path. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, this endpoint information is used to configure a pseudowire mesh to support L2VPN-based services.

The BGP autodiscovery mechanism facilitates the configuration of L2VPN services, which are an integral part of the VPLS feature. VPLS enables flexibility in deploying services by connecting geographically dispersed sites as a large LAN over high-speed Ethernet in a robust and scalable IP Multiprotocol Label Switching

(MPLS) network. For more information about BGP and the L2VPN address family in relation to VPLS Autodiscovery, see the following chapters in the *IP Routing: BGP Configuration Guide*:

- “BGP Support for the L2VPN Address Family” chapter

How Enabling VPLS Autodiscovery Differs from Manually Configuring VPLS

With VPLS Autodiscovery enabled, you no longer need to manually set up Virtual Private LAN Service (VPLS). The commands that you use to set up VPLS Autodiscovery are similar to those that you use to manually configure VPLS, as shown in the table below. VPLS Autodiscovery uses **neighbor** commands in L2VPN address family mode to distribute endpoint information to configure a pseudowire.

Table 34: Manual VPLS Configuration Versus VPLS Autodiscovery Configuration

Manual Configuration of VPLS	VPLS Autodiscovery BGP Based

`l2 vfi vpls1 manual`
` vpn id 100`
` neighbor 10.10.10.1 encapsulation mpls`
` neighbor 10.10.10.0 encapsulation mpls`
` exit`
	`l2 vfi vpls1 autodiscovery`
	` vpn id 100`
	` exit`
	`router bgp 1`
	` no bgp default ipv4-unicast`
	` bgp log-neighbor-changes`
	` bgp update-delay 1`
	` neighbor 10.1.1.2 remote-as 1`
	` neighbor 10.1.1.2 update-source Loopback1`
	` .`
	` .`
	` .`
	` address-family l2vpn vpls`
	` neighbor 10.1.1.2 activate`
	` neighbor 10.1.1.2 send-community extended`
	`exit-address-family`

Configure VPLS Autodiscovery by using the **l2 vfi autodiscovery** command. This command allows a virtual forwarding instance (VFI) to learn and advertise pseudowire endpoints. As a result, you no longer need to enter the **neighbor** command in L2 VFI configuration mode.

However, the **neighbor** command is still supported with VPLS Autodiscovery in L2 VFI configuration mode. You can use the **neighbor** command to allow PE devices that do not participate in the autodiscovery process to join the VPLS domain. You can also use the **neighbor** command with PE devices that have been configured using the Tunnel Selection feature. In addition, you can use the **neighbor** command in hierarchical VPLS configurations that have user-facing PE (U-PE) devices that do not participate in the autodiscovery process and have split-horizon forwarding disabled.

How Enabling VPLS Autodiscovery Differs from Manually Configuring VPLS using the commands associated with the L2VPN Protocol-Based CLIs feature

With VPLS Autodiscovery enabled, you no longer need to manually set up Virtual Private LAN Service (VPLS). The commands that you use to set up VPLS Autodiscovery are similar to those that you use to manually configure VPLS, as shown in the table below. VPLS Autodiscovery uses **neighbor** commands in L2VPN address family mode to distribute endpoint information to configure a pseudowire.

Table 35: Manual VPLS Configuration Versus VPLS Autodiscovery Configuration

<table>
<tr><th>Manual Configuration of VPLS</th><th>VPLS Autodiscovery BGP Based</th></tr>
<tr><td><pre>l2vpn vfi context vpls1
 vpn id 100
 neighbor 10.10.10.1 encapsulation mpls
 neighbor 10.10.10.0 encapsulation mpls
 exit</pre></td><td><pre>l2vpn vfi context vpls1
 vpn id 100
 autodiscovery bgp signaling ldp
 exit
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
 .
 .
 .
 address-family l2vpn vpls
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
exit-address-family</pre></td></tr>
</table>

Configure VPLS Autodiscovery by using the **autodiscovery** command. This command allows a virtual forwarding instance (VFI) to learn and advertise pseudowire endpoints. As a result, you no longer need to enter the **neighbor** command in L2 VFI configuration mode.

However, the **neighbor** command is still supported with VPLS Autodiscovery in L2 VFI configuration mode. You can use the **neighbor** command to allow PE devices that do not participate in the autodiscovery process to join the VPLS domain. You can also use the **neighbor** command with PE devices that have been configured using the Tunnel Selection feature. In addition, you can use the **neighbor** command in hierarchical VPLS configurations that have user-facing PE (U-PE) devices that do not participate in the autodiscovery process and have split-horizon forwarding disabled.

show Commands Affected by VPLS Autodiscovery BGP Based

The following **show** commands were enhanced for VPLS Autodiscovery:

- The **show mpls l2transport vc detail** command was updated to include Forwarding Equivalence Class (FEC) 129 signaling information for autodiscovered Virtual Private LAN Service (VPLS) pseudowires.
- The **show vfi** command was enhanced to display information related to autodiscovered virtual forwarding instances (VFIs). The new output includes the VPLS ID, the route distinguisher (RD), the route target (RT), and router IDs of discovered peers.
- The **show xconnect** command was updated with the **rib** keyword to provide Routing Information Base (RIB) information about pseudowires.

BGP VPLS Autodiscovery Support on a Route Reflector

By default, routes received from an internal BGP (iBGP) peer are not sent to another iBGP peer unless a full mesh configuration is formed between all BGP devices within an autonomous system (AS). This results in scalability issues. Using Border Gateway Protocol (BGP) route reflectors leads to much higher levels of scalability. Configuring a route reflector allows a device to advertise or reflect the iBGP learned routes to other iBGP speakers.

Virtual Private LAN Service (VPLS) Autodiscovery supports BGP route reflectors. A BGP route reflector can be used to reflect BGP VPLS prefixes without VPLS being explicitly configured on the route reflector.

A route reflector does not participate in autodiscovery; that is, no pseudowires are set up between the route reflector and the PE devices. A route reflector reflects VPLS prefixes to other PE devices so that these PE devices do not need to have a full mesh of BGP sessions. The network administrator configures only the BGP VPLS address family on a route reflector. For an example configuration of VPLS Autodiscovery support on a route reflector, see the “Example: BGP VPLS Autodiscovery Support on Route Reflector” section.

N-PE Access to VPLS Using MST

When a Virtual Private LAN Service (VPLS) network uses multihoming (network-facing PE [N-PE] VPLS redundancy) to prevent a single point of failure of an N-PE device, a bridging loop is introduced. One of the N-PE devices can be set as a Multiple Spanning Tree (MST) root to break the loop. In most cases, the two N-PE devices are also separated by a distance that makes direct physical link impossible. You can configure a virtual link (usually through the same VPLS core network) between the two N-PE devices to pass an MST bridge protocol data unit (BPDU) for path calculation, break the loop, and maintain convergence. The virtual link is created using a special pseudowire between the active and redundant N-PE devices.

While setting up an MST topology for a VPLS PE device, ensure the following:

- The **spanning-tree mode mst** command is enabled on all PE devices (N-PE and user-facing PE [U-PE]) participating in the MST topology.
- A special pseudowire is configured between the two N-PE devices, and these two devices are in the up state.
- The special pseudowire is a manually created virtual forwarding instance (VFI).
- The configuration (inlcuding the MST instance, the Ethernet virtual circuit [EVC], and the VLAN) on all PE devices is the same.
- One of the N-PE devices, and not one of the U-PE devices, is the root for the MST instance.
- The name and revision for the MST configuration are configured to synchronize with the standby Route Processor (RP).

How to Configure VPLS Autodiscovery BGP Based

Enabling VPLS Autodiscovery BGP Based

Perform this task to enable Virtual Private LAN Service (VPLS) PE devices to discover other PE devices that are part of the same VPLS domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *vfi-name* **autodiscovery**
4. **vpn id** *vpn-id*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *vfi-name* **autodiscovery** **Example:** `Device(config)# l2 vfi vpls1 autodiscovery`	Enables VPLS Autodiscovery on a PE device and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits L2 VFI configuration mode.

Enabling VPLS Autodiscovery BGP Based using the commands associated with the L2VPN Protocol-Based CLIs feature

Perform this task to enable Virtual Private LAN Service (VPLS) PE devices to discover other PE devices that are part of the same VPLS domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling** {**ldp** | **bgp**}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls1`	Establishes an L2VPN VFI context and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling** {**ldp** | **bgp**} **Example:** `Device(config-vfi)# autodiscovery bgp signaling ldp`	Enables the VPLS Autodiscovery: BGP Based feature on the PE device.
Step 6	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits L2 VFI configuration mode.

Configuring VPLS BGP Signaling

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling** {**bgp** | **ldp**} [**template** *template-name*]
6. **ve id** *ve-id*
7. **ve range** *ve-range*
8. **exit**
9. **exit**
10. **router bgp** *autonomous-system-number*
11. **bgp graceful-restart**
12. **neighbor** *ip-address* **remote-as** *autonomous-system-number*

13. **address-family l2vpn** [**vpls**]
14. **neighbor** *ip-address* **activate**
15. **neighbor** *ip-address* **send-community** [**both** | **standard** | **extended**]
16. **neighbor** *ip-address* **suppress-signaling-protocol ldp**
17. **end**
18. **show bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *name* **Example:** `Device(config)# l2vpn vfi context vfi1`	Establishes a L2VPN virtual forwarding interface (VFI) between two or more separate networks and enters Layer 2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 100`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling** {**bgp** | **ldp**} [**template** *template-name*] **Example:** `Device(config-vfi)# autodiscovery bgp signaling bgp`	Enables BGP signaling and discovery or LDP signaling and enters L2VPN VFI autodiscovery configuration mode. **Note** For the VPLS BGP Signaling feature use the **autodiscovery bgp signaling bgp** command.
Step 6	**ve id** *ve-id* **Example:** `Device(config-vfi-autodiscovery)# ve id 1001`	Specifies the VPLS endpoint (VE) device ID value. The VE ID identifies a VFI within a VPLS service. The VE device ID value is from 1 to 16384.
Step 7	**ve range** *ve-range* **Example:** `Device(config-vfi-autodiscovery)# ve range 12`	Specifies the VE device ID range value. The VE range overrides the minimum size of VE blocks. The default minimum size is 10. Any configured VE range must be higher than 10.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config-vfi-autodiscovery)# exit`	Exits L2VPN VFI autodiscovery configuration mode and enters L2VPN VFI configuration mode.
Step 9	**exit** **Example:** `Device(config-vfi)# exit`	Exits L2VPN VFI configuration mode and enters global configuration mode.
Step 10	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode to create or configure a BGP routing process.
Step 11	**bgp graceful-restart** **Example:** `Device(config-router)# bgp graceful-restart`	Enables the BGP graceful restart capability and BGP nonstop forwarding (NSF) awareness.
Step 12	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 remote-as 100`	Configures peering with a BGP neighbor in the specified autonomous system.
Step 13	**address-family l2vpn** [**vpls**] **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The optional **vpls** keyword specifies that VPLS endpoint provisioning information is to be distributed to BGP peers. In this example, an L2VPN VPLS address family session is created.
Step 14	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 activate`	Enables the neighbor to exchange information for the L2VPN VPLS address family with the local device.
Step 15	**neighbor** *ip-address* **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 10.10.10.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.

	Command or Action	Purpose
Step 16	**neighbor** *ip-address* **suppress-signaling-protocol ldp** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 suppress-signaling-protocol ldp`	Suppresses LDP signaling and enables BGP signaling. • In this example LDP signaling is suppressed (and BGP signaling enabled) for the neighbor at 10.10.10.1.
Step 17	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 18	**show bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*} **Example:** `Device# show bgp l2vpn vpls all`	(Optional) Displays information about the L2VPN VPLS address family.

Configuring BGP to Enable VPLS Autodiscovery

The Border Gateway Protocol (BGP) Layer 2 VPN (L2VPN) address family supports a separate L2VPN Routing Information Base (RIB) that contains endpoint provisioning information for Virtual Private LAN Service (VPLS) Autodiscovery. BGP learns the endpoint provisioning information from the L2VPN database, which is updated each time a Layer 2 virtual forwarding instance (VFI) is configured. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to configure a pseudowire mesh to support L2VPN-based services.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **no bgp default ipv4-unicast**
5. **bgp log-neighbor-changes**
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number*
8. Repeat Steps 6 and 7 to configure other BGP neighbors.
9. **address-family l2vpn** [**vpls**]
10. **neighbor** {*ip-address* | *peer-group-name*} **activate**
11. **neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**}
12. Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.
13. **exit-address-family**
14. **end**
15. **show vfi**
16. **show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured using the **neighbor remote-as** router configuration command unless you configure the **no bgp default ipv4-unicast** router configuration command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.
Step 5	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 remote-as 65000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor at 10.10.10.1 is an internal BGP neighbor.

	Command or Action	Purpose
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 update-source loopback1`	(Optional) Configures a device to select a specific source or interface to receive routing table updates. • This example uses a loopback interface. The advantage of this configuration is that the loopback interface is not affected by the effects of a flapping interface.
Step 8	Repeat Steps 6 and 7 to configure other BGP neighbors.	—
Step 9	**address-family l2vpn** [**vpls**] **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The optional **vpls** keyword specifies that the VPLS endpoint provisioning information is to be distributed to BGP peers. • In this example, an L2VPN VPLS address family session is created.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**} **Example:** `Device(config-router-af)# neighbor 10.10.10.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.
Step 12	Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.	—
Step 13	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.
Step 14	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.
Step 15	**show vfi** **Example:** `Device# show vfi`	Displays information about the configured VFI instances.
Step 16	**show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*} **Example:**	Displays information about the L2VPN VPLS address family.

	Command or Action	Purpose
	`Device# show ip bgp l2vpn vpls all`	

Customizing the VPLS Autodiscovery Settings

Several commands allow you to customize the Virtual Private LAN Service (VPLS) environment. You can specify identifiers for the VPLS domain, the route distinguisher (RD), the route target (RT), and the provider edge (PE) device. Perform this task to customize these identifiers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *vfi-name* **autodiscovery**
4. **vpn id** *vpn-id*
5. **vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
6. **rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
7. **route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
8. **auto-route-target**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *vfi-name* **autodiscovery** **Example:** `Device(config)# l2 vfi vpls1 autodiscovery`	Enables VPLS Autodiscovery on the PE device and enters Layer 2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# vpls-id 5:300`	(Optional) Assigns an identifier to the VPLS domain. • This command is optional because VPLS Autodiscovery automatically generates a VPLS ID using the Border Gateway Protocol (BGP) autonomous system (AS) number and the configured VFI VPN ID.

	Command or Action	Purpose
		You can use this command to change the automatically generated VPLS ID. • There are two formats for configuring the VPLS ID argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 6	**rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# rd 2:3`	(Optional) Specifies the RD to distribute endpoint information. • This command is optional because VPLS Autodiscovery automatically generates an RD using the BGP autonomous system number and the configured VFI VPN ID. You can use this command to change the automatically generated RD. • There are two formats for configuring the route distinguisher argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 7	**route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# route-target 600:2222`	(Optional) Specifies the RT. • This command is optional because VPLS Autodiscovery automatically generates an RT using the lower 6 bytes of the RD and the VPLS ID. You can use this command to change the automatically generated RT. • There are two formats for configuring the route target argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 8	**auto-route-target** **Example:** `Device(config-vfi)# auto-route-target`	(Optional) Enables the automatic generation of a RT.
Step 9	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits Layer 2 VFI configuration mode.

Configuring BGP to Enable VPLS Autodiscovery using the commands associated with the L2VPN Protocol-Based CLIs feature

The BGP L2VPN address family supports a separate L2VPN Routing Information Base (RIB) that contains endpoint provisioning information for Virtual Private LAN Service (VPLS) Autodiscovery. BGP learns the endpoint provisioning information from the L2VPN database, which is updated each time a Layer 2 virtual forwarding instance (VFI) is configured. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to configure a pseudowire mesh to support L2VPN-based services.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **no bgp default ipv4-unicast**
5. **bgp log-neighbor-changes**
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number*
8. Repeat Steps 6 and 7 to configure other BGP neighbors.
9. **address-family l2vpn** [**vpls**]
10. **neighbor** {*ip-address* | *peer-group-name*} **activate**
11. **neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**}
12. Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.
13. **exit-address-family**
14. **end**
15. **show l2vpn vfi**
16. **show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 4	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured using the **neighbor remote-as** router configuration command unless you configure the **no bgp default ipv4-unicast** router configuration command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.
Step 5	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 remote-as 65000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor at 10.10.10.1 is an internal BGP neighbor.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 update-source loopback1`	(Optional) Configures a device to select a specific source or interface to receive routing table updates. • This example uses a loopback interface. The advantage of this configuration is that the loopback interface is not affected by the effects of a flapping interface.
Step 8	Repeat Steps 6 and 7 to configure other BGP neighbors.	—
Step 9	**address-family l2vpn** [**vpls**] **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The optional **vpls** keyword specifies that the VPLS endpoint provisioning information is to be distributed to BGP peers.

	Command or Action	Purpose
		• In this example, an L2VPN VPLS address family session is created.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**} **Example:** `Device(config-router-af)# neighbor 10.10.10.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.
Step 12	Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.	—
Step 13	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.
Step 14	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.
Step 15	**show l2vpn vfi** **Example:** `Device# show l2vpn vfi`	Displays information about the Layer 2 VPN (L2VPN) virtual forwarding instances (VFI).
Step 16	**show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*} **Example:** `Device# show ip bgp l2vpn vpls all`	Displays information about the L2VPN VPLS address family.

Customizing the VPLS Autodiscovery Settings using the commands associated with the L2VPN Protocol-Based CLIs feature

Several commands allow you to customize the Virtual Private LAN Service (VPLS) environment. You can specify identifiers for the VPLS domain, the route distinguisher (RD), the route target (RT), and the provider edge (PE) device. Perform this task to customize these identifiers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling** {**ldp** | **bgp**}
6. **vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
7. **rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
8. **route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
9. **auto-route-target**
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls1`	Establishes a L2VPN VFI context and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling** {**ldp** | **bgp**} **Example:** `Device(config-vfi)# autodiscovery bgp signaling ldp`	Enables the VPLS Autodiscovery: BGP Based feature on the PE device.
Step 6	**vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# vpls-id 5:300`	(Optional) Assigns an identifier to the VPLS domain. • This command is optional because VPLS Autodiscovery automatically generates a VPLS ID using the Border Gateway Protocol (BGP) autonomous system (AS) number and the configured

	Command or Action	Purpose
		VFI VPN ID. You can use this command to change the automatically generated VPLS ID. • There are two formats for configuring the VPLS ID argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 7	**rd** {*autonomous-system-number***:***nn* | *ip-address***:***nn*} **Example:** `Device(config-vfi)# rd 2:3`	(Optional) Specifies the RD to distribute endpoint information. • This command is optional because VPLS Autodiscovery automatically generates an RD using the BGP autonomous system number and the configured VFI VPN ID. You can use this command to change the automatically generated RD. • There are two formats for configuring the route distinguisher argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 8	**route-target** [**import** | **export** | **both**] {*autonomous-system-number***:***nn* | *ip-address***:***nn*} **Example:** `Device(config-vfi)# route-target 600:2222`	(Optional) Specifies the RT. • This command is optional because VPLS Autodiscovery automatically generates an RT using the lower 6 bytes of the RD and the VPLS ID. You can use this command to change the automatically generated RT. • There are two formats for configuring the route target argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 9	**auto-route-target** **Example:** `Device(config-vfi)# auto-route-target`	(Optional) Enables the automatic generation of a RT.
Step 10	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits Layer 2 VFI configuration mode.

Configuring MST on VPLS N-PE Devices

A network-facing PE (N-PE) device is the root bridge for a Multiple Spanning Tree (MST) instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *vfi-name* **manual**
4. **vpn id** *vpn-id*
5. **forward permit l2protocol all**
6. **neighbor** *peer-N-PE-ip-address* **encapsulation mpls**
7. **exit**
8. **spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**]
9. **spanning-tree mst configuration**
10. **name** *name*
11. **revision** *version*
12. **instance** *instance-id* **vlan** *vlan-range*
13. **end**
14. **show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *vfi-name* **manual** **Example:** `Device(config)# l2 vfi vpls-mst manual`	Creates a Layer 2 virtual forwarding instance (VFI) and enters Layer 2 VFI manual configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 4000`	Sets or updates the VPN ID on a VPN routing and forwarding (VRF) instance.
Step 5	**forward permit l2protocol all** **Example:** `Device(config-vfi)# forward permit l2protocol all`	Defines the VPLS pseudowire that is used to transport the bridge protocol data unit (BPDU) information between two N-PE devices.

	Command or Action	Purpose
Step 6	**neighbor** *peer-N-PE-ip-address* **encapsulation mpls** **Example:** `Device(config-vfi)# neighbor 10.76.100.12 encapsulation mpls`	Specifies the type of tunnel signaling and encapsulation mechanism for each VPLS peer.
Step 7	**exit** **Example:** `Device(config-vfi)# exit`	Exits Layer 2 VFI manual configuration mode and returns to global configuration mode.
Step 8	**spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**] **Example:** `Device(config)# spanning-tree mode mst`	Switches between MST, Per-VLAN Spanning Tree+ (PVST+), and Rapid-PVST+ modes.
Step 9	**spanning-tree mst configuration** **Example:** `Device(config)# spanning-tree mst configuration`	Enters MST configuration mode.
Step 10	**name** *name* **Example:** `Device(config-mst)# name cisco`	Sets the name for the MST region.
Step 11	**revision** *version* **Example:** `Device(config-mst)# revision 11`	Sets the revision number for the MST configuration.
Step 12	**instance** *instance-id* **vlan** *vlan-range* **Example:** `Device(config-mst)# instance 1 vlan 100`	Maps a VLAN or a group of VLANs to an MST instance.
Step 13	**end** **Example:** `Device(config-mst)# end`	Exits MST configuration mode and enters privileged EXEC mode.
Step 14	**show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]] **Example:** `Device# show spanning-tree mst 1`	Displays information about the MST configuration.

Configuring MST on VPLS N-PE Devices using the commands associated with the L2VPN Protocol-Based CLIs feature

A network-facing PE (N-PE) device is the root bridge for a Multiple Spanning Tree (MST) instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **forward permit l2protocol all**
6. **neighbor** *peer-N-PE-ip-address* **encapsulation mpls**
7. **exit**
8. **spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**]
9. **spanning-tree mst configuration**
10. **name** *name*
11. **revision** *version*
12. **instance** *instance-id* **vlan** *vlan-range*
13. **end**
14. **show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls-mst`	Establishes an L2VPN VFI context and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 4000`	Sets or updates the VPN ID on a VPN routing and forwarding (VRF) instance.
Step 5	**forward permit l2protocol all** **Example:** `Device(config-vfi)# forward permit l2protocol all`	Defines the VPLS pseudowire that is used to transport the bridge protocol data unit (BPDU) information between two N-PE devices.
Step 6	**neighbor** *peer-N-PE-ip-address* **encapsulation mpls** **Example:**	Specifies the type of tunnel signaling and encapsulation mechanism for each VPLS peer.

	Command or Action	Purpose
	`Device(config-vfi)# neighbor 10.76.100.12 encapsulation mpls`	
Step 7	**exit** **Example:** `Device(config-vfi)# exit`	Exits Layer 2 VFI manual configuration mode and returns to global configuration mode.
Step 8	**spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**] **Example:** `Device(config)# spanning-tree mode mst`	Switches between MST, Per-VLAN Spanning Tree+ (PVST+), and Rapid-PVST+ modes.
Step 9	**spanning-tree mst configuration** **Example:** `Device(config)# spanning-tree mst configuration`	Enters MST configuration mode.
Step 10	**name** *name* **Example:** `Device(config-mst)# name cisco`	Sets the name for the MST region.
Step 11	**revision** *version* **Example:** `Device(config-mst)# revision 11`	Sets the revision number for the MST configuration.
Step 12	**instance** *instance-id* **vlan** *vlan-range* **Example:** `Device(config-mst)# instance 1 vlan 100`	Maps a VLAN or a group of VLANs to an MST instance.
Step 13	**end** **Example:** `Device(config-mst)# end`	Exits MST configuration mode and enters privileged EXEC mode.
Step 14	**show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]] **Example:** `Device# show spanning-tree mst 1`	Displays information about the MST configuration.

Configuration Examples for VPLS Autodiscovery BGP Based

The following examples show the configuration of a network that uses VPLS Autodiscovery:

Example: Enabling VPLS Autodiscovery BGP Based

```
Device> enable
Device# configure terminal
Device(config)# l2 vfi vpls1 autodiscovery
Device(config-vfi)# vpn id 10
Device(config-vfi)# exit
```

Example: Enabling VPLS Autodiscovery BGP Based Using Commands Associated with L2VPN Protocol-Based Feature

```
Device> enable
Device# configure terminal
Device(config)# l2vpn vfi context vpls1
Device(config-vfi)# vpn id 10
Device(config-vfi)# autodiscovery bgp signaling ldp
Device(config-vfi)# exit
```

Example: Configuring BGP to Enable VPLS Autodiscovery

PE1

```
l2 router-id 10.1.1.1
l2 vfi auto autodiscovery
 vpn id 100
!
pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.1 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.1 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
```

```
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE2

```
l2 router-id 10.1.1.2
l2 vfi auto autodiscovery
 vpn id 100
!
 pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.2 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.2 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE3

```
l2 router-id 10.1.1.3
l2 vfi auto autodiscovery
 vpn id 100
```

```
!
pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.3 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.3 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 exit-address-family
```

Example: Configuring BGP to Enable VPLS Autodiscovery Using Commands Associated with L2VPN Protocol-Based Feature

PE1

```
l2vpn
 router-id 10.1.1.1
l2vpn vfi context auto
 vpn id 100
 autodiscovery bgp signaling ldp
!
interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
!
interface Loopback1
 ip address 10.1.1.1 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.1 255.255.255.0
 mpls ip
!
```

```
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE2

```
l2vpn
 router-id 10.1.1.2
l2vpn vfi context auto
 vpn id 100
 autodiscovery bgp signaling ldp

!
 interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
!
interface Loopback1
 ip address 10.1.1.2 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.2 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
```

```
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE3

```
l2vpn
 router-id 10.1.1.3
l2vpn vfi context auto
 vpn id 100
 autodiscovery bgp signaling ldp

!
interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
!
interface Loopback1
 ip address 10.1.1.3 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.3 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 exit-address-family
```

Example: Customizing VPLS Autodiscovery Settings

```
Device> enable
Device# configure terminal
Device(config)# l2 vfi vpls1 autodiscovery
```

```
Device(config-vfi)# vpn id 10
Device(config-vfi)# vpls-id 5:300
Device(config-vfi)# rd 2:3
Device(config-vfi)# route-target 600:2222
Device(config-vfi)# end
```

Example: Customizing VPLS Autodiscovery Settings using the commands associated with the L2VPN Protocol-Based CLIs feature

```
Device> enable
Device# configure terminal
Device(config)# l2vpn vfi context vpls1
Device(config-vfi)# vpn id 10
Device(config-vfi)# autodiscovery bgp signaling ldp
Device(config-vfi)# vpls-id 5:300
Device(config-vfi)# rd 2:3
Device(config-vfi)# route-target 600:2222
Device(config-vfi)# end
```

Example: Configuring MST on VPLS N-PE Devices

```
Device> enable
Device# configure terminal
Device(config)# l2 vfi vpls-mst manual
Device(config-vfi)# vpn id 4000
Device(config-vfi)# forward permit l2protocol all
Device(config-vfi)# neighbor 10.76.100.12 encapsulation mpls
Device(config-vfi)# exit
Device(config)# spanning-tree mode mst
Device(config)# spanning-tree mst configuration
Device(config-mst)# name cisco
Device(config-mst)# revision 11
Device(config-mst)# instance 1 vlan 100
Device(config-mst)# end
```

The following is sample output from the **show spanning-tree mst** command:

```
Device# show spanning-tree mst 1

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1                              // Root for MST instance
 1 with VLAN 100
Interface                       Role Sts Cost      Prio.Nbr Type
-----------                 --- -- ------ ---- ------------------------------
Gi1/0/0                         Desg FWD 20000     128.18   P2p   // Access interface
VPLS-MST                        Desg FWD 1         128.28   Shr   // Forward VFI
```

The following is sample output from the **show spanning-tree mst detail** command:

```
Device# show spanning-tree mst 1 detail

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1           // Root for MST instance 1 with VLAN 100
GigabitEthernet1/0/0 of MST1 is designated forwarding
Port info             port id         128.18  priority    128  cost       20000
```

```
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.18
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 40, received 5
VPLS-4000 of MST1 is designated forwarding
Port info             port id         128.28  priority    128  cost           1
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.28
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 28, received 26        // BPDU message exchange between N-PE devices
```

Example: Configuring MST on VPLS N-PE Devices using the commands associated with the L2VPN Protocol-Based CLIs feature

```
Device> enable
Device# configure terminal
Device(config)# l2vpn vfi context vpls-mst
Device(config-vfi)# vpn id 4000
Device(config-vfi)# forward permit l2protocol all
Device(config-vfi)# member 10.76.100.12 encapsulation mpls
Device(config-vfi)# exit
Device(config)# spanning-tree mode mst
Device(config)# spanning-tree mst configuration
Device(config-mst)# name cisco
Device(config-mst)# revision 11
Device(config-mst)# instance 1 vlan 100
Device(config-mst)# end
```

The following is sample output from the **show spanning-tree mst** command:

```
Device# show spanning-tree mst 1

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1                               // Root for MST instance
 1 with VLAN 100
Interface                         Role Sts Cost      Prio.Nbr Type
-----------                  --- -- ------ ---- ------------------------------
Gi1/0/0                           Desg FWD 20000     128.18   P2p   // Access interface
VPLS-MST                          Desg FWD 1         128.28   Shr   // Forward VFI
```

The following is sample output from the **show spanning-tree mst detail** command:

```
Device# show spanning-tree mst 1 detail

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1            // Root for MST instance 1 with VLAN 100
GigabitEthernet1/0/0 of MST1 is designated forwarding
Port info             port id         128.18  priority    128  cost       20000
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.18
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 40, received 5
VPLS-4000 of MST1 is designated forwarding
Port info             port id         128.28  priority    128  cost           1
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.28
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 28, received 26        // BPDU message exchange between N-PE devices
```

Example: BGP VPLS Autodiscovery Support on Route Reflector

In the following example, a host named PE-RR (indicating Provider Edge-Route Reflector) is configured as a route reflector that is capable of reflecting Virtual Private LAN Service (VPLS) prefixes. The VPLS address family is configured using the **address-family l2vpn vpls** command.

```
hostname PE-RR
!
router bgp 1
 bgp router-id 10.1.1.3
 no bgp default route-target filter
 bgp log-neighbor-changes
neighbor iBGP-PEERS peer-group
neighbor iBGP-PEERS remote-as 1
neighbor iBGP-PEERS update-source Loopback1
neighbor 10.1.1.1 peer-group iBGP-PEERS
neighbor 10.1.1.2 peer-group iBGP-PEERS
!
address-family l2vpn vpls
  neighbor iBGP-PEERS send-community extended
  neighbor iBGP-PEERS route-reflector-client
  neighbor 10.1.1.1 peer-group iBGP-PEERS
  neighbor 10.1.1.2 peer-group iBGP-PEERS
exit-address-family
```

Additional References for VPLS Autodiscovery BGP Based

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference

Standards and RFCs

Standard/RFC	Title
draft-ietf-l2vpn-signaling-08.txt	*Provisioning, Autodiscovery, and Signaling in L2VPNs*
draft-ietf-l2vpn-vpls-bgp-08.8	*Virtual Private LAN Service (VPLS) Using BGP for Autodiscovery and Signaling*
draft-ietf-mpls-lsp-ping-03.txt	*Detecting MPLS Data Plane Failures*
draft-ietf-pwe3-vccv-01.txt	*Pseudo-Wire (PW) Virtual Circuit Connection Verification (VCCV)*
RFC 3916	*Requirements for Pseudo-wire Emulation Edge-to-Edge (PWE3)*
RFC 3981	*Pseudo Wire Emulation Edge-to-Edge Architecture*
RFC 6074	Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)

Standard/RFC	Title
RFC 4761	Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling

MIBs

MIB	MIBs Link
• CISCO-IETF-PW-ATM-MIB (PW-ATM-MIB) • CISCO-IETF-PW-ENET-MIB (PW-ENET-MIB) • CISCO-IETF-PW-FR-MIB (PW-FR-MIB) • CISCO-IETF-PW-MIB (PW-MIB) • CISCO-IETF-PW-MPLS-MIB (PW-MPLS-MIB)	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register on Cisco.com.	http://www.cisco.com/techsupport

Feature Information for VPLS Autodiscovery BGP Based

Table 36: Feature Information for VPLS Autodiscovery BGP Based

Feature Name	Releases	Feature Information
VPLS Autodiscovery BGP Based	Cisco IOS XE Release 3.7S Cisco IOS Release 15.1(1)SY	VPLS Autodiscovery enables each Virtual Private LAN Service (VPLS) provider edge (PE) device to discover other PE devices that are part of the same VPLS domain.

CHAPTER 23

N:1 PVC Mapping to PWE with Nonunique VPIs

The N:1 PVC Mapping to PseudoWire Emulation (PWE) with Nonunique virtual path identifiers (VPIs) feature maps one or more ATM permanent virtual circuits (PVCs) to a single pseudowire (PW). There are two modes of AAL0 encapsulation, N:1 and 1:1 mapping. In N:1 mapping, multiple unrelated virtual path identifier/virtual channel identifier (VPI/VCI) are carried over a single Multiprotocol Label Switching (MPLS) PW. This is an efficient mapping method because less resources are used from the MPLS network. In 1:1 mapping, a single VPI/VCI is carried over a single MPLS PW. Benefits of this feature include the following:

- Aggregate quality of service (QoS) can be applied to related PVCs.
- Bandwidth is conserved with the reduction in the number of pseudowires that are used.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for N:1 PVC Mapping to PWE with Nonunique VPIs

- N:1 permanent virtual circuits (PVC) mapping configuration is supported only on multipoint subinterfaces; it is not supported on main interfaces or point-to-point subinterfaces.
- N:1 PVC mapping mode is not supported on Access Circuit Redundancy subinterfaces.

- Preconfigured PVCs cannot exist on the multipoint subinterface on which you want to configure N:1 PVC mapping.
- An attachment circuit that has been bound to a pseudowire cannot be removed unless all Layer 2 virtual circuits (VCs) have been removed.
- Layer 3 PVCs cannot be configured on N:1 subinterfaces.
- Cell packing values configured under a VC class attached to the PVC, main interface, or subinterface will not be inherited by N:1 PVCs.
- Operation, Administration, and Maintenance (OAM) functionality is not supported on N:1 Layer 2 PVCs. OAM cells coming from the customer edge (CE) network will be treated as normal data traffic and will traverse through the pseudowire.
- Only ATM adaptation layer type 0 (AAL0) encapsulation is supported for N:1 PVCs.
- The service policy configuration can be configured only at the subinterface level for N:1 PVCs.

Information About N:1 PVC Mapping to PWE with Nonunique VPIs

N:1 PVC Mapping to PWE with Nonunique VPIs Feature Description

To transport ATM cells over Multiprotocol Label Switching (MPLS), a VC is established between the provider edge (PE) routers on both ends of the MPLS backbone. With the N:1 permanent virtual circuit (PVC) Mapping to PseudoWire Emulation (PWE) with Nonunique VPIs feature, multiple PVCs irrespective of their Virtual Path Identifiers (VPIs), are transported over a single pseudowire configured on a subinterface. ("N:1" refers to the number of PVCs transported over one pseudowire). ATM cells are packed together in a single frame and sent over the single pseudowire. The ATM cell header information is packed together with the cell payload on a per-cell basis in the packets so that packets received at the egress end are unpacked and the ATM cells are mapped to the respective PVCs.

In N:1 PVC mapping mode, the device can pack cells only from a single PVC in an MPLS packet to transmit over a pseudowire; cells from multiple PVCs cannot be packed in a single MPLS packet and mapped to a single pseudowire for transmission. However, if a device receives an MPLS packet that is packed with cells from multiple PVCs, then those cells will be unpacked and sent to the respective PVCs.

How to Configure N:1 PVC Mapping to PWE with Nonunique VPIs

Configuring N:1 PVC Mapping to PWE with Nonunique VPIs

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **interface atm** *slot/subslot/port*
4. **atm mcpt-timers** *timer1 timer2 timer3*
5. **exit**
6. **configure terminal**
7. **interface atm** *slot/subslot/port.subslot* **multipoint**
8. **no ip address**
9. **atm enable-ilmi-trap**
10. **cell-packing** *maxcells* **mcpt-timer** *timer-number*
11. **xconnect** *peer-ipaddress* *vc-id* **encapsulation** **mpls**
12. **pvc** *vpi/vci* **l2transport**
13. Repeat Step 12 for the number of PVCs that you want to configure.
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot/subslot/port* **Example:** `Device(config)# interface atm 9/1/1`	Enables the ATM interface and enters interface configuration mode.
Step 4	**atm mcpt-timers** *timer1 timer2 timer3* **Example:** `Device(config-if)# atm mcpt-timers 100 200 300`	Sets the Maximum Cell Packing Timeout (MCPT) values in microseconds. • The MCPT timer sets the time for which the device waits for the raw cells (AAL0 encapsulation) to be packed into a single packet for punting to the pseudowire.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 6	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 7	**interface atm** *slot/subslot/port.subslot* **multipoint** **Example:** `Device(config)# interface atm 9/1/1.1 multipoint`	Enters subinterface configuration mode and creates a multipoint subinterface on the given port on the specified ATM Shared Port Adapter (SPA).
Step 8	**no ip address** **Example:** `Device(config-subif)# no ip address`	Removes the interface IP address.
Step 9	**atm enable-ilmi-trap** **Example:** `Device(config-subif)# atm enable-ilmi-trap`	Generates an Integrated Local Management Interface (ILMI) atmfVccChange trap when an ATM interface or subinterface is enabled or shut down.
Step 10	**cell-packing** *maxcells* **mcpt-timer** *timer-number* **Example:** `Device(config-subif)# cell-packing 20 mcpt-timer 2`	Enables ATM over MPLS to pack multiple ATM cells into each MPLS packet within the MCPT timing.
Step 11	**xconnect** *peer-ipaddress* *vc-id* **encapsulation mpls** **Example:** `Device(config-subif)# xconnect 10.1.1.1 100 encapsulation mpls`	(Optional) Enables the attachment circuit and specifies the IP address of the peer, a VC ID, and the data encapsulation method.
Step 12	**pvc** *vpi/vci* **l2transport** **Example:** `Device(config-subif)# pvc 10/100 l2transport`	Assigns a VPI and virtual channel identifier (VCI).
Step 13	Repeat Step 12 for the number of PVCs that you want to configure.	—
Step 14	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Configuring N:1 PVC Mapping to PWE with Nonunique VPIs using the commands associated with the L2VPN Protocol-Based CLIs feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot/subslot/port*
4. **atm mcpt-timers** *timer1 timer2 timer3*
5. **exit**

6. **configure terminal**
7. **interface atm** *slot/subslot/portt.subslot* **multipoint**
8. **no ip address**
9. **atm enable-ilmi-trap**
10. **cell-packing** *maxcells* **mcpt-timer** *timer-number*
11. **end**
12. **interface pseudowire** *number*
13. **encapsulation mpls**
14. **neighbor** *peer-address vcid-value*
15. **exit**
16. **l2vpn xconnect context** *context-name*
17. **member pseudowire** *interface-number*
18. **member gigabitethernet** *interface-number*
19. **end**
20. **pvc** *vpi/vci* **l2transport**
21. Repeat Step 12 for the number of PVCs that you want to configure.
22. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface atm** *slot/subslot/port* **Example:** `Device(config)# interface atm 9/1/1`	Enables the ATM interface and enters interface configuration mode.
Step 4	**atm mcpt-timers** *timer1 timer2 timer3* **Example:** `Device(config-if)# atm mcpt-timers 100 200 300`	Sets the Maximum Cell Packing Timeout (MCPT) values in microseconds. • The MCPT timer sets the time for which the device waits for the raw cells (AAL0 encapsulation) to be packed into a single packet for punting to the pseudowire.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 6	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 7	**interface atm** *slot/subslot/portt.subslot* **multipoint** **Example:** `Device(config)# interface atm 9/1/1.1 multipoint`	Enters subinterface configuration mode and creates a multipoint subinterface on the given port on the specified ATM Shared Port Adapter (SPA).
Step 8	**no ip address** **Example:** `Device(config-subif)# no ip address`	Removes the interface IP address.
Step 9	**atm enable-ilmi-trap** **Example:** `Device(config-subif)# atm enable-ilmi-trap`	Generates an Integrated Local Management Interface (ILMI) atmfVccChange trap when an ATM interface or subinterface is enabled or shut down.
Step 10	**cell-packing** *maxcells* **mcpt-timer** *timer-number* **Example:** `Device(config-subif)# cell-packing 20 mcpt-timer 2`	Enables ATM over MPLS to pack multiple ATM cells into each MPLS packet within the MCPT timing.
Step 11	**end** **Example:** `Router(config-subif)# end`	Exits to privileged EXEC mode.
Step 12	**interface pseudowire** *number* **Example:** `Router(config)# interface pseudowire 100`	Specifies the pseudowire interface and enters interface configuration mode.
Step 13	**encapsulation mpls** **Example:** `Router(config-if)# encapsulation mpls`	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method.
Step 14	**neighbor** *peer-address vcid-value* **Example:** `Router(config-if)# neighbor 10.1.1.1 100`	Specifies the peer IP address and virtual circuit (VC) ID value of the Layer 2 VPN (L2VPN) pseudowire.
Step 15	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 16	**l2vpn xconnect context** *context-name* **Example:** `Router(config)# l2vpn xconnect context con1`	Creates a Layer 2 VPN (L2VPN) cross connect context and enters xconnect configuration mode.
Step 17	**member pseudowire** *interface-number* **Example:** `Router(config-xconnect)# member pseudowire 100`	Specifies a member pseudowire to form a Layer 2 VPN (L2VPN) cross connect.
Step 18	**member gigabitethernet** *interface-number* **Example:** `Router(config-xconnect)# member GigabitEthernet0/0/0.1`	Specifies the location of the Gigabit Ethernet member interface.
Step 19	**end** **Example:** `Router(config-xconnect)# end`	Exits to privileged EXEC mode.
Step 20	**pvc** *vpi/vci* **l2transport** **Example:** `Device(config-subif)# pvc 10/100 l2transport`	Assigns a VPI and virtual channel identifier (VCI).
Step 21	Repeat Step 12 for the number of PVCs that you want to configure.	—
Step 22	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Configuration Examples for N:1 PVC Mapping to PWE with Nonunique VPIs

Example: Configuring N:1 PVC Mapping to PWE with Nonunique VPIs

The following example shows how to configure the N:1 ATM permanent virtual circuit (PVC) mapping to pseudowires with non unique virtual path identifiers (VPIs):

```
Device> enable
Device# configure terminal
Device(config)# interface atm 9/1/1
Device(config-if)# atm mcpt-timers 500 5000 50000
```

```
Device(config-if)# exit
Device# configure terminal
Device(config)# interface atm 9/1/1.1 multipoint
Device(config-subif)# no ip address
Device(config-subif)# atm enable-ilmi-trap
Device(config-subif)# cell packing 20 mcpt-timer 2
Device(config-subif)# xconnect 10.1.1.1 100 encapsulation mpls
Device(config-subif)# pvc 10/100 l2transport
Device(config-subif)# pvc 11/122 l2transport
Device(config-subif)# pvc 19/231 l2transport
Device(config-subif)# end
```

Example: Configuring N:1 PVC Mapping to PWE with Nonunique VPIs using the commands associated with the L2VPN Protocol-Based CLIs feature

The following example shows how to configure the N:1 ATM permanent virtual circuit (PVC) mapping to pseudowires with non unique virtual path identifiers (VPIs):

```
Device> enable
Device# configure terminal
Device(config)# interface atm 9/1/1
Device(config-if)# atm mcpt-timers 500 5000 50000
Device(config-if)# exit
Device(config)# configure terminal
Device(config)# interface atm 9/1/1.1 multipoint
Device(config-subif)# no ip address
Device(config-subif)# atm enable-ilmi-trap
Device(config-subif)# cell packing 20 mcpt-timer 2
Device(config-subif)# exit
Device(config)#interface pseudowire 100
Device(config-if)#encapsulation mpls
Device(config-if)#neighbor 10.1.1.1 100
Device(config-if)# pvc 10/100 l2transport
Device(config-if)# pvc 11/122 l2transport
Device(config-if)# pvc 19/231 l2transport
Device(config-if)# exit
Device(config)#l2vpn xconnect context A
Router(config-xconnect)#member pseudowire 100
Device(config-xconnect)#member atm 9/1/1
Device(config-xconnect)# end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List
ATM commands	Asynchronous Transfer Mode Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for N:1 PVC Mapping to PWE with Nonunique VPIs

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 37: Feature Information for N:1 PVC Mapping to PWE with Nonunique VPIs

Feature Name	Releases	Feature Information
N:1 PVC Mapping to PWE with Nonunique VPIs	Cisco IOS XE Release 3.7S	The N:1 PVC Mapping to PWE with Nonunique VPIs feature maps one or more ATM PVCs to a single pseudowire. In Cisco IOS XE Release 3.7S, support was added for Cisco ASR 903 Routers. The following command was introduced by this feature: **show atm cell-packaging** .

CHAPTER 24

QoS Policies for VFI Pseudowires

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for QoS Policies for VFI Pseudowires

- A maximum of 32K pseudowires.
- A maximum of 4K unique policy maps.
- A maximum of 128 neighbors per VFI context.

Information About QoS Policies for VFI Pseudowires

QoS Policies for VFI Pseudowires

QoS policies are specified on individual pseudowire interfaces and are applied only to the corresponding pseudowires. It is possible to specify different QoS policies on different pseudowire members of the same virtual forwarding interface (VFI) or on the subset of the pseudowires. There may be one or more pseudowires configured per VFI. Both manually configured and auto discovered pseudowire configurations are supported.

QoS policies are specified using a pseudowire template. The template can be applied on multiple pseudowires of the same, or different, VFIs. All those pseudowires get the same QoS policy applied as specified in the template. For auto-discovered pseudowires, QoS policies can only be specified using a pseudowire template.

The QoS Policies for VFI Pseudowires feature supports both ingress and egress policies and traffic classification can be done based on different match criteria.

How to Configure QoS Policies for VFI Pseudowires

Configuring QoS Policies for Pseudowires

Perform this task to configure QoS policies for pseudowires.

Before you begin

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-map-name*
5. **priority** *bandwidth-kbps*
6. **exit**
7. **class** *class-map-name*
8. **bandwidth percent** *percentage*
9. **exit**
10. **class** *class-map-name*
11. **police cir** *bps*
12. **exit**
13. **class** *class-map-name*
14. **shape average** *bps*
15. **queue-limit** *queue-limit size* **packets**
16. **random-detect**
17. **exit**
18. **exit**
19. **policy-map** *policy-map-name*
20. **class** *class-map-name*
21. **shape average** *bps*
22. **service-policy** *policy-map*
23. **exit**
24. **exit**
25. **policy-map** *policy-map-name*
26. **class** *class-map-name*

27. **shape average** *bps*
28. **exit**
29. **exit**
30. **policy-map** *policy-map-name*
31. **class** *class-map-name*
32. **shape average** *bps*
33. **exit**
34. **exit**
35. **exit policy-map** *policy-map-name*
36. **class** *class-map-name*
37. **shape average** *bps*
38. **exit**
39. **exit**
40. **policy-map** *policy-map-name*
41. **class** *class-map-name*
42. **police** *bps*
43. **interface pseudowire** *number*
44. **encap mpls**
45. **neighbor** *peer-address vcid-value*
46. **service-policy input** *policy-map-name*
47. **service-policy output** *policy-map-name*
48. **interface gigabit ethernet** *number*
49. **service-policy output** *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **Note** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device# policy-map gold-policy-child`	Creates a policy map to specify a service policy.
Step 4	**class** *class-map-name* **Example:** `Device(config-pmap)# class priority-class`	Specifies the name of the class map.

	Command or Action	Purpose
Step 5	**priority** *bandwidth-kbps* **Example:** `Device(config-pmap-c)# priority 100`	Gives priority to a class of traffic belonging to a policy map.
Step 6	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 7	**class** *class-map-name* **Example:** `Device(config-pmap-c)# class guarantee-class`	Specifies the name of the class map.
Step 8	**bandwidth percent** *percentage* **Example:** `Device(config-pmap-c)# bandwidth percent 50`	Specifies or modifies the bandwidth allocated for a class belonging to a policy map.
Step 9	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 10	**class** *class-map-name* **Example:** `Device(config-pmap-c)# class limited-class`	Specifies the name of the class map.
Step 11	**police cir** *bps* **Example:** `Device(config-pmap-c)# police cir 8000`	Creates a per-interface policer and configures the policy-map class to use it.
Step 12	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 13	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class map.
Step 14	**shape average** *bps* **Example:**	Shapes traffic to the indicated bit rate.

	Command or Action	Purpose
	`Device(config-pmap-c)# shape average 8000`	
Step 15	**queue-limit** *queue-limit size* **packets** **Example:** `Device(config-pmap-c)# queue-limit 150 packets`	Specifies the queue limit size for a class.
Step 16	**random-detect** **Example:** `Device(config-pmap-c)# andom-detect`	Configures Weighted Random Early Detection (WRED) for a class in a policy map.
Step 17	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 18	**exit** **Example:** `Device(config-pmap)# exit`	Exits policy-map configuration mode.
Step 19	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map gold-policy-hqos`	Creates a policy map to specify a service policy.
Step 20	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class map.
Step 21	**shape average** *bps* **Example:** `Device(config-pmap-c)# shape average 10000`	Shapes traffic to the indicated bit rate.
Step 22	**service-policy** *policy-map* **Example:** `Device(config-pmap-c)# service-policy gold-policy-child`	Attaches a policy map to a class.
Step 23	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.

	Command or Action	Purpose
Step 24	**exit** **Example:** `Device(config-pmap)# exit`	Exits policy-map configuration mode.
Step 25	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map pw-shaper`	Creates a policy map to specify a service policy.
Step 26	**class** *class-map-name* **Example:** `Device(config-pmap)#class class-default`	Specifies the name of the class map.
Step 27	**shape average** *bps* **Example:** `Device(config-pmap-c)#shape average 20000`	Shapes traffic to the indicated bit rate.
Step 28	**exit** **Example:** `Device(config-pmap-c)#exit`	Exits policy-map class configuration mode.
Step 29	**exit** **Example:** `Device(config-pmap)#exit`	Exits policy-map configuration mode.
Step 30	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map sub-ifc-shaper`	Creates a policy map to specify a service policy.
Step 31	**class** *class-map-name* **Example:** `Device(config-pmap)#class class-default`	Specifies the name of the class map.
Step 32	**shape average** *bps* **Example:** `Device(config-pmap-c)#shape average 40000`	Shapes traffic to the indicated bit rate.
Step 33	**exit** **Example:**	Exits policy-map class configuration mode.

	Command or Action	Purpose
	`Device(config-pmap-c)#exit`	
Step 34	**exit** **Example:** `Device(config-pmap)#exit`	Exits policy-map configuration mode.
Step 35	**exit policy-map** *policy-map-name* **Example:** `Device(config)# policy-map port-shaper`	Creates a policy map to specify a service policy.
Step 36	**class** *class-map-name* **Example:** `Device(config-pmap)#class class-default`	Specifies the name of the class map.
Step 37	**shape average** *bps* **Example:** `Device(config-pmap-c)#shape average 60000`	Shapes traffic to the indicated bit rate.
Step 38	**exit** **Example:** `Device(config-pmap-c)#exit`	Exits policy-map class configuration mode.
Step 39	**exit** **Example:** `Device(config-pmap)#exit`	Exits policy-map configuration mode.
Step 40	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map ingress-police`	Creates a policy map to specify a service policy.
Step 41	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	
Step 42	**police** *bps* **Example:** `Device(config-pmap-c)# police 10000`	Creates a per-interface policer and configures the policy-map class to use it.

	Command or Action	Purpose
Step 43	**interface pseudowire** *number* **Example:** `Device(config-pmap-c-police)# interface pseudowire 1`	Configures an interface type and enters interface configuration mode.
Step 44	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 45	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.0.0.1 100`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 46	**service-policy input** *policy-map-name* **Example:** `Device(config-if)# service-policy input ingress-policy`	Attaches a policy map to an input interface.
Step 47	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output gold-policy-hqos`	Attaches a policy map to an output interface.
Step 48	**interface gigabit ethernet** *number* **Example:** `Device(config-if)# interface gigabitethernet 1/1/0`	Configures an interface type.
Step 49	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output port-shaper`	Attaches a policy map to an output interface.

Creating a Hierarchical Policy for VFI Pseudowires

Perform this task to create a hierarchical policy for VFI Pseudowires.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **policy-map** *policy-map-name*
4. **class** *class-map-name*
5. **shape average** *bps*
6. **service-policy** *policy-map*
7. **exit**
8. **exit**
9. **policy-map** *policy-map-name*
10. **class** *class-map-name*
11. **shape average** *bps*
12. **exit**
13. **exit**
14. **policy-map** *policy-map-name*
15. **class** *class-map-name*
16. **shape average** *bps*
17. **exit**
18. **exit**
19. **exit policy-map** *policy-map-name*
20. **class** *class-map-name*
21. **shape average** *bps*
22. **exit**
23. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **Note** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map gold-policy-hqos`	Creates a policy map to specify a service policy.
Step 4	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class map.

	Command or Action	Purpose
Step 5	**shape average** *bps* **Example:** `Device(config-pmap-c)# shape average 10000`	Shapes traffic to the indicated bit rate.
Step 6	**service-policy** *policy-map* **Example:** `Device(config-pmap-c)# service-policy gold-policy-child`	Attaches a policy map to a class.
Step 7	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 8	**exit** **Example:** `Device(config-pmap)# exit`	Exits policy-map configuration mode.
Step 9	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map pw-shaper`	Creates a policy map to specify a service policy.
Step 10	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class map.
Step 11	**shape average** *bps* **Example:** `Device(config-pmap-c)# shape average 20000`	Shapes traffic to the indicated bit rate.
Step 12	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 13	**exit** **Example:** `Device(config-pmap)# exit`	Exits policy-map configuration mode.

	Command or Action	Purpose
Step 14	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map sub-ifc-shaper`	Creates a policy map to specify a service policy.
Step 15	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class map.
Step 16	**shape average** *bps* **Example:** `Device(config-pmap-c)# shape average 40000`	Shapes traffic to the indicated bit rate.
Step 17	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 18	**exit** **Example:** `Device(config-pmap)# exit`	Exits policy-map configuration mode.
Step 19	**exit policy-map** *policy-map-name* **Example:** `Device(config)# policy-map port-shaper`	Creates a policy map to specify a service policy.
Step 20	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class map.
Step 21	**shape average** *bps* **Example:** `Device(config-pmap-c)# shape average 60000`	Shapes traffic to the indicated bit rate.
Step 22	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 23	**exit** **Example:**	Exits policy-map configuration mode.

	Command or Action	Purpose
	`Device(config-pmap)# exit`	

Attaching a Policy Map to a VFI Pseudowire

Perform this task to attach a policy map to a VFI Pseudowire.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-map-name*
5. **police** *bps*
6. **interface pseudowire** *number*
7. **encap mpls**
8. **neighbor** *peer-address vcid-value*
9. **service-policy input** *policy-map-name*
10. **service-policy output** *policy-map-name*
11. **interface gigabit ethernet** *number*
12. **service-policy output** *policy-map-name*
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **Note** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device# policy-map ingress-police`	Creates a policy map to specify a service policy.
Step 4	**class** *class-map-name* **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class map.

	Command or Action	Purpose
Step 5	**police** *bps* **Example:** `Device(config-pmap-c)# police 10000`	Creates a per-interface policer and configures the policy-map class to use it.
Step 6	**interface pseudowire** *number* **Example:** `Device(config-pmap-c-police)# interface pseudowire 1`	Configures an interface type and enters interface configuration mode.
Step 7	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 8	**neighbor** *peer-address vcid-value* **Example:** `Device(config-if)# neighbor 10.0.0.1 100`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 9	**service-policy input** *policy-map-name* **Example:** `Device(config-if)# service-policy input ingress-policy`	Attaches a policy map to an input interface.
Step 10	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output gold-policy-hqos`	Attaches a policy map to an output interface.
Step 11	**interface gigabit ethernet** *number* **Example:** `Device(config-if)# interface gigabit ethernet 1/1/0`	Configures an interface type.
Step 12	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output port-shaper`	Attaches a policy map to an output interface.
Step 13	**exit** **Example:**	Exits interface configuration mode.

	Command or Action	Purpose
	`Device(config-if)# exit`	

Configuring VFI with Two Pseudowire Members with Different QoS Policies

Perform this task to configure VFI with two pseudowire members with different QoS policies.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface pseudowire** *number*
4. **encap mpls**
5. **neighbor** *peer-address vcid value*
6. **service-policy output** *policy-map-name*
7. **interface pseudowire** *number*
8. **encap mpls**
9. **neighbor** *peer-address vcid value*
10. **service-policy output** *policy-map-name*
11. **l2vpn vfi context** *name*
12. **vpn id** *vpn-id*
13. **member pseudowire** *pw-int-number*
14. **member pseudowire** *pw-int-number*
15. **bridge-domain** *bridge-domain-id*
16. **member** *interface-type-number*
17. **interface BDI** *number*
18. **ip vrf forwarding** *vrf-name*
19. **ip address** *ip-address mask*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **Note** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface pseudowire** *number* **Example:**	Configures an interface type and enters interface configuration mode.

	Command or Action	Purpose
	`Device# interface pseudowire 1`	
Step 4	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 5	**neighbor** *peer-address vcid value* **Example:** `Device(config-if)# neighbor 10.0.0.1 100`	Specifies the peer IP address and virtual circuit (VC) ID value of an L2VPN pseudowire.
Step 6	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output gold-policy`	Attaches a policy map to an output interface.
Step 7	**interface pseudowire** *number* **Example:** `Device(config-if)# interface pseudowire 2`	Configures an interface type.
Step 8	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 9	**neighbor** *peer-address vcid value* **Example:** `Device(config-if)# neighbor 20.0.0.1 100`	Specifies the peer IP address and VCID of an L2VPN pseudowire.
Step 10	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output silver-policy`	Attaches a policy map to an output interface.
Step 11	**l2vpn vfi context** *name* **Example:** `Device(config-if)# l2vpn vfi context my-vfi`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) between two or more separate networks.
Step 12	**vpn id** *vpn-id* **Example:**	Sets a VPN ID on a Virtual Private LAN Services (VPLS) instance.

	Command or Action	Purpose
	`Device(config-vfi)# vpn id 100`	
Step 13	**member pseudowire** *pw-int-number* **Example:** `Device(config-vfi)# member pseudowire 1`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection.
Step 14	**member pseudowire** *pw-int-number* **Example:** `Device(config-vfi)# member pseudowire 2`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection.
Step 15	**bridge-domain** *bridge-domain-id* **Example:** `Device(config-vfi)# bridge-domain 100`	Configures components on a bridge domain.
Step 16	**member** *interface-type-number* **Example:** `Device(config-bdomain)# member vfi my-vfi`	Binds a service instance to a bridge domain instance.
Step 17	**interface BDI** *number* **Example:** `Device(config-bdomain)# interface BDI 100`	Configures an interface type and enters interface configuration mode.
Step 18	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding MY-VRF`	Associates a Virtual Private Network (VPN) routing and forwarding (VRF) instance with an interface or subinterface.
Step 19	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 30.0.0.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.

Configuring VFI with Two Pseudowire Members with the Same QoS Policy

Perform this task to configure VFI with two pseudowire members with the same QoS policy.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **template type pseudowire** *name*
4. **encap mpls**

5. **service-policy output** *policy-map-name*
6. **interface pseudowire** *number*
7. **encap mpls**
8. **neighbor** *peer-address vcid value*
9. **source template type pseudowire** *template-name*
10. **interface pseudowire** *number*
11. **encap mpls**
12. **neighbor** *peer-address vcid value*
13. **source template type pseudowire** *template-name*
14. **l2vpn vfi context** *name*
15. **vpn id** *vpn-id*
16. **member pseudowire** *pw-int-number*
17. **member pseudowire** *pw-int-number*
18. **bridge-domain** *bridge-domain-id*
19. **member** *interface-type-number*
20. **interface BDI** *number*
21. **ip vrf forwarding** *vrf-name*
22. **ip address** *ip-address mask*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **Note** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** *name* **Example:** `Device(config)# template type pseudowire my_template`	Configures a template.
Step 4	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 5	**service-policy output** *policy-map-name* **Example:**	Attaches a policy map to a output interface.

	Command or Action	Purpose
	`Device(config-template)# service-policy output common-policy`	
Step 6	**interface pseudowire** *number* **Example:** `Device(config-if)# interface pseudowire 1`	Configures an interface type.
Step 7	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 8	**neighbor** *peer-address vcid value* **Example:** `Device(config-if)# neighbor 10.0.0.1 100`	Specifies the peer IP address and VCID of an L2VPN pseudowire.
Step 9	**source template type pseudowire** *template-name* **Example:** `Device(config-if)# source template type pseudowire my_template`	Configures the name of a source template of type pseudowire.
Step 10	**interface pseudowire** *number* **Example:** `Device(config-if)# interface pseudowire 2`	Configures an interface type.
Step 11	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 12	**neighbor** *peer-address vcid value* **Example:** `Device(config-if)# neighbor 20.0.0.1 100`	Specifies the peer IP address and VCID of an L2VPN pseudowire.
Step 13	**source template type pseudowire** *template-name* **Example:** `Device(config-if)# source template type pseudowire my_template`	Configures the name of a source template of type pseudowire.
Step 14	**l2vpn vfi context** *name* **Example:** `Device(config-if)# l2vpn vfi context my-vfi`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) between two or more separate networks.

	Command or Action	Purpose
Step 15	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 100`	Sets a VPN ID on a Virtual Private LAN Services (VPLS) instance.
Step 16	**member pseudowire** *pw-int-number* **Example:** `Device(config-vfi)# member pseudowire 1`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection.
Step 17	**member pseudowire** *pw-int-number* **Example:** `Device(config-vfi)# member pseudowire 2`	Specifies the devices that form a point-to-point Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) connection.
Step 18	**bridge-domain** *bridge-domain-id* **Example:** `Device(config-vfi)# bridge-domain 100`	Configures components on a bridge domain.
Step 19	**member** *interface-type-number* **Example:** `Device(config-bdomain)# member vfi my-vfi`	Binds a service instance to a bridge domain instance.
Step 20	**interface BDI** *number* **Example:** `Device(config-bdomain)# interface BDI 100`	Configures an interface type and enters interface configuration mode.
Step 21	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding MY-VRF`	Associates a Virtual Private Network (VPN) routing and forwarding (VRF) instance with an interface or subinterface.
Step 22	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 30.0.0.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.

Configuring VFI with Auto Discovered Pseudowires

Perform this task to configure VFI with auto discovered pseudowires.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **template type pseudowire** *name*
4. **encap mpls**
5. **service-policy output** *policy-map-name*
6. **l2vpn vfi context** *name*
7. **vpn id** *vpn-id*
8. **autodiscovery bgp signaling ldp template** *template-name*
9. **bridge-domain** *bridge-domain-id*
10. **member** *interface-type-number*
11. **interface BDI** *number*
12. **ip vrf forwarding** *vrf-name*
13. **ip address** *ip-address mask*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **Note** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template type pseudowire** *name* **Example:** `Device(config)# template type pseudowire my_template`	Configures a template.
Step 4	**encap mpls** **Example:** `Device(config-if)# encap mpls`	Configures MPLS encapsulation.
Step 5	**service-policy output** *policy-map-name* **Example:** `Device(config-template)# service-policy output common-policy`	Attaches a policy map to a output interface.
Step 6	**l2vpn vfi context** *name* **Example:** `Device(config-if)# l2vpn vfi context my-vfi`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) between two or more separate networks.

	Command or Action	Purpose
Step 7	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 100`	Sets a VPN ID on a Virtual Private LAN Services (VPLS) instance.
Step 8	**autodiscovery bgp signaling ldp template** *template-name* **Example:** `Device(config-vfi)# autodiscovery bgp signaling ldp template my_template`	Designates a Layer 2 virtual forwarding interface (VFI) as having Label Distribution Protocol (LDP) autodiscovered pseudowire members.
Step 9	**bridge-domain** *bridge-domain-id* **Example:** `Device(config-vfi)# bridge-domain 100`	Configures components on a bridge domain.
Step 10	**member** *interface-type-number* **Example:** `Device(config-bdomain)# member vfi my-vfi`	Binds a service instance to a bridge domain instance.
Step 11	**interface BDI** *number* **Example:** `Device(config-bdomain)# interface BDI 100`	Configures an interface type and enters interface configuration mode.
Step 12	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding MY-VRF`	Associates a Virtual Private Network (VPN) routing and forwarding (VRF) instance with an interface or subinterface.
Step 13	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 30.0.0.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.

Configuration Examples for QoS Policies for VFI Pseudowires

Example: Configuring QoS Policies for Pseudowires

The following example shows how to QoS policies for pseudowires:

```
Device(config)# policy-map GOLD-POLICY-CHILD
Device(config-pmap)# class PRIORITY-CLASS
```

```
Device(config-pmap-c)# priority 100
Device(config-pmap-c)# exit
Device(config-pmap)# class GUARANTEE-CLASS
Device(config-pmap-c)# bandwidth 1000
Device(config-pmap-c)# exit
Device(config-pmap)# class LIMITED-CLASS
Device(config-pmap-c)# police cir 8000
Device(config-pmap-c-police)# class class-default
Device(config-pmap-c)# shape average 8000
Device(config-pmap-c)# queue-limit 150
Device(config-pmap-c)# random-detect
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# policy-map GOLD-POLICY-HQOS
Device(config-pmap)# class class-default
Device(config-pmap-c)# shape average 8000
Device(config-pmap-c)# service-policy GOLD-POLICY-CHILD
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# policy-map PW-SHAPER
Device(config-pmap)# class class-default
Device(config-pmap-c)# shape average 8000
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# policy-map SUB-IFC-SHAPER
Device(config-pmap)# class class-default
Device(config-pmap-c)# shape average 10000
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# policy-map PORT-SHAPER
Device(config-pmap)# class class-default
Device(config-pmap-c)# shape average 20000
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# policy-map INGRESS-POLICE
Device(config-pmap)# class class-default
Device(config-pmap-c)# police 10000
Device(config-pmap-c-police)# interface pseudowire 1
Line protocol on Interface pseudowire0, changed state to up
Device(config-if)# encap mpls
Device(config-if)# neighbor 10.0.0.1 100
Device(config-if)# service-policy input INGRESS-POLICY
Device(config-if)# service-policy output GOLD-POLICY-HQOS
Device(config-if)# interface GigabitEthernet 1/1/0
--- Pseudowire is going out through this interface
Device(config-if)# service-policy output PORT-SHAPER
```

Example: Configuring VFI with Two Pseudowire Members with Different QoS Policies

The following example shows how to configure VFI with two pseudowire members with different QoS policies:

```
Device(config)# interface pseudowire1
Line protocol on Interface pseudowire0, changed state to up
Device(config-if)# encap mpls
Device(config-if)# neighbor 10.0.0.1 100
```

```
Device(config-if)# service-policy output GOLD-POLICY
Device(config-if)# interface pseudowire2
Device(config-if)# encap mpls
Device(config-if)# neighbor 20.0.0.1 100
Device(config-if)# service-policy output SILVER-POLICY
Device(config-if)# l2vpn vfi context MY-VFI
Device(config-vfi)# vpn id 100
Device(config-vfi)# member pseudowire1
Device(config-vfi)# member pseudowire2
Device(config-vfi)# bridge-domain 100
Device(config-bdomain)# member vfi MY-VFI
STATUS_CHANGED: Status of VFI my-vfi changed from DOWN to UP
Device(config-bdomain)# interface BDI 100
Device(config-if)# ip vrf forwarding MY-VRF
Device(config-if)# ip address 30.0.0.1 255.255.255.0
```

Example: Configuring VFI with Two Pseudowire Members with the Same QoS Policy

The following example shows how to configure VFI with two pseudowire members with the same QoS policy:

```
Device(config)# template type pseudowire MY_TEMPLATE
Device(config-template)# encapsulation mpls
Device(config-template)# service-policy output COMMON-POLICY
Device(config-template)# interface pseudowire1
Line protocol on Interface pseudowire0, changed state to up
Device(config-if)# encap mpls
Device(config-if)# neighbor 10.0.0.1 100
Device(config-if)# source template type pseudowire MY_TEMPLATE
Device(config-if)# interface pseudowire2
Device(config-if)# encap mpls
Device(config-if)# neighbor 20.0.0.1 100
Device(config-if)# source template type pseudowire MY_TEMPLATE
Device(config-if)# l2vpn vfi context MY-VFI
Device(config-vfi)# vpn id 100
Device(config-vfi)# member pseudowire1
Device(config-vfi)# member pseudowire2
Device(config-vfi)# bridge-domain 100
Device(config-bdomain)# member vfi MY-VFI
Status of VFI my-vfi changed from DOWN to UP
Device(config-bdomain)# interface BDI 100
Device(config-if)# ip vrf forwarding MY-VRF
Device(config-if)# ip address 30.0.0.1 255.255.255.0
```

Example: Configuring VFI with Auto Discovered Pseudowires

The following example shows how to configure VFI with auto discovered pseudowires:

```
Device(config)# template type pseudowire MY_TEMPLATE
Device(config-template)# encapsulation mpls
Device(config-template)# service-policy output COMMON-POLICY
Device(config-template)# l2vpn vfi context MY-VFI
Device(config-vfi)# vpn id 100
```

```
Line protocol on Interface pseudowire0, changed state to up
Device(config-vfi)# autodiscovery bgp signaling ldp template MY_TEMPLATE
Device(config-vfi-autodiscovery)# bridge-domain 100
Device(config-bdomain)# member vfi MY-VFI
Status of VFI my-vfi changed from DOWN to UP
Device(config-bdomain)# interface BDI 100
Device(config-if)# ip vrf forwarding MY-VRF
Device(config-if)# ip address 30.0.0.1 255.255.255.0
```

Example: Displaying Pseudowire Policy Map Information

The following is sample output from the **show policy-map interface** command which shows class maps and policy maps configured for the pseudowire 2 interface:

```
Device#show policy-map interface pseudowire2
 pseudowire2

 Service-policy output: pw_brr

    Class-map: prec1 (match-all)
      0 packets, 0 bytes
      30 second offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 1
      Queueing
      queue limit 4166 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      bandwidth remaining ratio 1

    Class-map: prec2 (match-all)
      0 packets, 0 bytes
      30 second offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 2
      Queueing
      queue limit 4166 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      bandwidth remaining ratio 2

    Class-map: prec3 (match-all)
      0 packets, 0 bytes
      30 second offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 3
      Queueing
      queue limit 4166 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      bandwidth remaining ratio 3

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      30 second offered rate 0000 bps, drop rate 0000 bps
      Match: any
      Queueing
      queue limit 4166 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      bandwidth remaining ratio 4
Device#
```

Additional References for QoS Policies for VFI Pseudowires

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	*Cisco IOS Multiprotocol Label Switching Command Reference*
QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*
Configuring the pseudowire class	"Any Transport over MPLS"
Layer 2 VPN	• Any Transport over MPLS • L2VPN Pseudowire Switching • MPLS LSP Ping/Traceroute for LDP/TE, and LSP Ping for VCCV
L2VPN pseudowires	• L2VPN Pseudowire Redundancy • MPLS Pseudowire Status Signaling

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information For QoS Policies for VFI Pseudowires

Table 38: Feature Information for QoS Policies for VFI Pseudowire

Feature Name	Releases	Feature Information
QoS Policies for VFI Pseudowires	Cisco IOS XE 3.8S	This features allows you to configure QoS classes and policies for use on VFI pseudowire members. The following commands were introduced or modified: **show policy-map interface**.

CHAPTER 25

VPLS BGP Signaling L2VPN Inter-AS Option A

The Virtual Private LAN Switching (VPLS) Border Gateway Protocol (BGP) Signaling Layer 2 Virtual Private Network (L2VPN) feature simplifies the auto-discovery and signaling of all known PE devices in a VPLS instance by using BGP.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for VPLS BGP Signaling L2VPN Inter-AS Option A

- The Control word must turned off for VPLS BGP signaling by using the **no control-word** command under a pseudowire class. For example:

```
Router> enable
Router# configure terminal
Router(config)# pseudowire-class my_pw_class
Router(config-pw-class)# no control-word
```

- The Route Distinguisher (RD) must match for all the virtual forwarding instances (VFIs) in a VPLS domain.

Information About VPLS BGP Signaling L2VPN Inter-AS Option A

BGP Auto-discovery and Signaling for VPLS

The Virtual Private LAN Switching (VPLS) control plane is used for auto-discovery and signaling. Auto-discovery involves locating all provider edge (PE) devices that participate in a particular VPLS instance. Signaling is accomplished by configuring pseudowires for a VPLS instance. Prior to the introduction of the VPLS BGP Signaling L2VPN Inter-AS Option B feature, Label Distribution Protocol (LDP) was used for signaling and Border Gateway Protocol (BGP) was used for auto-discovery, as specified in RFC 6074. With the introduction of the VPLS BGP Signaling L2VPN Inter-AS Option B feature, the VPLS BGP Signaling L2VPN feature supports RFC 4761 by simplifying the auto-discovery and signaling of all known PE devices in a VPLS instance by using BGP for both functions. Auto-discovery is defined per VPLS instance.

Internal BGP (IBGP) peers exchange update messages of the L2VPN Address Family Identifier (AFI) and the Subsequent Address Family Identifier (SAFI) numbers with L2VPN information to perform both auto-discovery and signaling, which includes the Network Layer Reachability Information (NLRI).

Both BGP standards (RFC 6074 and RFC 4761) for the auto-discovery protocol for VPLS use the same BGP AFI (25) and SAFI (65) but they have different Network Layer Reachability Information (NLRI) encoding, which makes them incompatible with each other. CLI configuration is needed to distinguish the two encoding types as they are mutually exclusive per neighbor. The difference between the two BGP standards is:

- RFC 6074 provides guidelines for specifying length encoding as bits.
- RFC 4761 provides guidelines for specifying length encoding as bytes.

To detect which NLRI encoding standard is supported, the length encoding needs to be determined.

BGP L2VPN Signaling with NLRI

Network Layer Reachability Information (NLRI) enables Border Gateway Protocol (BGP) to carry supernetting information, as well as perform aggregation. Each NLRI consists of block labels that follow the structure LB, LB+1, ... , LB+VBS-1. The NLRI is exchanged between BGP devices for BGP auto-discovery with BGP signaling. The following fields are configured or auto-generated for each Virtual Private LAN Switching (VPLS) instance:

- Length (2 Octets)
- Route distinguisher (RD) is usually an auto-generated 8-byte VPN ID that can also be configured. This value must be unique for a VPLS bridge-domain (or instance).
- VPLS Endpoint ID (VEID) (2 Octets). Each PE device is configured with a VEID value.
- VPLS Endpoint Block Offset (VBO) (2 Octets).
- VPLS Endpoint Block Size (VBS) (2 Octets).
- Label Base (LB) (3 Octets).

- Extended Community Type (2 Octets) - 0x800A attributes. The Route Target (RT) specified for a VPLS instance, next-hop and other Layer 2 information is carried in this encoding. An RT-based import and export mechanism similar to L3VPN is performed by BGP to perform filtering on the L2VPN NLRIs of a particular VPLS instance.
- Encapsulation Type (1 Octet) - VPLS = 19
- Control Flags (1 Octet)
- Layer 2 Maximum Transmission Unit (MTU) (2 Octets)
- Reserved (2 Octets)

How to Configure VPLS BGP Signaling L2VPN Inter-AS Option A

Enabling BGP Auto-discovery and BGP Signaling

Perform this task to enable Virtual Private LAN Service (VPLS) PE devices to discover other PE devices by BGP auto-discovery and BGP signaling functions announced through IBGP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-context-name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling bgp**
6. **ve id** *ve-ID-number*
7. **ve range** *ve-range-number*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-context-name* **Example:**	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) for specifying core-facing pseudowires in

	Command or Action	Purpose
	`Device(config)# l2vpn vfi context vfi1`	a Virtual Private LAN Services (VPLS) and enters L2VFI configuration mode. • The VFI represents an emulated LAN or a VPLS forwarder from the VPLS architectural model when using an emulated LAN interface.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling bgp** **Example:** `Device(config-vfi)# autodiscovery bgp signaling bgp`	Enables BGP auto-discovery and BGP signaling on the device.
Step 6	**ve id** *ve-ID-number* **Example:** `Device(config-vfi)# ve id 1`	Configures a VPLS Endpoint ID (VEID) for the NLRI exchanged between BGP devices for BGP auto-discovery with BGP signaling. • For example, VEID numbering sequences such as 1,2,3 or 501, 502, 503 are preferred because the VEIDs are contiguous. • Avoid a non-contiguous numbering scheme such as 100, 200, 300. Repeat this step to add more VEIDs. The VEID must be unique within the same VPLS domain for all PE devices. **Note** If you change the VEID, then the virtual circuit (VC) reprovisions and traffic is impacted as a result.
Step 7	**ve range** *ve-range-number* **Example:** `Device(config-vfi)# ve range 10`	Overrides the minimum size of VPLS edge (VE) blocks. • The VE range value should be approximately the same as the number of neighbors (up to 100). • The VE range can be configured based on the number of neighboring PE devices in the network. • For example, if 50 PE devices are in a VPLS domain, then a VE range of 50 is better than 10 because the number of NLRIs exchanged are less and the convergence time is reduced. **Note** If no VE range is configured or an existing VE range value is removed, then the default VE range of 10 is applied. The default VE range should not be used if the device has many PE neighbors. **Note** If you change the VE range, then the VC reprovisions and traffic is impacted as a result.

	Command or Action	Purpose
Step 8	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. **Note** Commands take effect after the device exits L2VFI configuration mode.

Configuring BGP Signaling for VPLS Autodiscovery

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp graceful-restart**
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
6. **address-family l2vpn vpls**
7. **neighbor** {*ip-address* | *peer-group-name*} **activate**
8. **neighbor** {*ip-address* | *peer-group-name*} **send-community extended**
9. **neighbor** {*ip-address* | *peer-group-name*} **suppress-signaling-protocol ldp**
10. **exit-address-family**
11. Repeat steps 1 to 10 to configure and activate other BGP neighbors in an L2VPN address family.
12. **end**
13. **show l2vpn vfi**
14. **show ip bgp l2vpn vpls** {**all** [**summary**] | **rd** *route-distinguisher*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.
Step 4	**bgp graceful-restart** **Example:** `Device(config-router)# bgp graceful-restart`	Enables the Border Gateway Protocol (BGP) graceful restart capability globally for all BGP neighbors.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 198.51.100.1 remote-as 65000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor at 10.10.10.1 is an internal BGP neighbor.
Step 6	**address-family l2vpn vpls** **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The **vpls** keyword specifies that the VPLS endpoint provisioning information is to be distributed to BGP peers and a L2VPN VPLS address family session is created.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 198.51.100.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **send-community extended** **Example:** `Device(config-router-af)# neighbor 198.51.100.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **suppress-signaling-protocol ldp** **Example:** `Device(config-router-af)# neighbor 198.51.100.1 suppress-signaling protocol ldp`	Suppresses LDP signaling for a BGP neighbor so that BGP signaling for VPLS auto-discovery is used instead. • In this example, LDP signaling is suppressed for the neighbor at 10.10.10.1.
Step 10	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.
Step 11	Repeat steps 1 to 10 to configure and activate other BGP neighbors in an L2VPN address family.	

	Command or Action	Purpose
Step 12	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

Step 13	**show l2vpn vfi**
	Example:
	`Device# show l2vpn vfi`
	
	`PE1-standby#sh l2vpn vfi`
	`Load for five secs: 0%/0%; one minute: 0%; five minutes: 0%`
	`Time source is hardware calendar, *20:50:52.526 GMT Wed Aug 29 2012`
	
	`Legend: RT=Route-target, S=Split-horizon, Y=Yes, N=No`
	
	`VFI name: VFI1, state: up, type: multipoint, signaling: BGP`
	`  VPN ID: 1, VE-ID: 10, VE-SIZE: 10`
	`  RD: 1:1, RT: 1:1`
	`  Bridge-Domain 100 attachment circuits:`
	`  Pseudo-port interface: pseudowire100001`
	`  Interface          Peer Address    VE-ID  Local Label  Remote Label    S`
	`  pseudowire100003   198.51.100.2     11     1003        2002           Y`
	`  pseudowire100005   198.51.100.3     12     1004        2002           Y`
	
	`VFI name: VFI2, state: up, type: multipoint, signaling: BGP`
	`  VPN ID: 2, VE-ID: 20, VE-SIZE: 12`
	`  RD: 1:2, RT: 1:2, import 3:3, export 4:4`
	`  Bridge-Domain 200 attachment circuits:`
	`  Pseudo-port interface: pseudowire100002`
	`  Interface          Peer Address    VE-ID  Local Label  Remote Label    S`
	`  pseudowire100004   198.51.100.2         21 1021         2020            Y`
	`  pseudowire100006   198.51.100.3         22 1022         2020            Y`
		Displays information about the configured VFI instances.

Step 14	**show ip bgp l2vpn vpls** {**all** [**summary**] | **rd** *route-distinguisher*}
	Example:
	`Device# show ip bgp l2vpn vpls all summary`
	
	`BGP router identifier 198.51.100.1, local AS number 65000`
	`BGP table version is 14743, main routing table version 14743`
	`6552 network entries using 1677312 bytes of memory`
	`6552 path entries using 838656 bytes of memory`
	`3276/3276 BGP path/bestpath attribute entries using 760032 bytes of memory`
	`1638 BGP extended community entries using 65520`
		Displays information about the L2VPN VPLS address family.

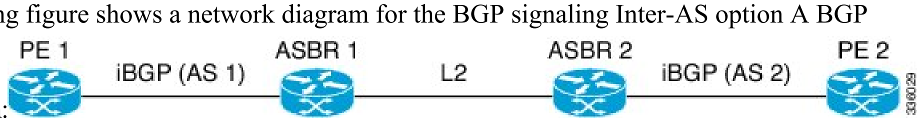
	Command or Action	Purpose
	bytes of memory 0 BGP route-map cache entries using 0 bytes of memory 0 BGP filter-list cache entries using 0 bytes of memory BGP using 3341520 total bytes of memory BGP activity 9828/3276 prefixes, 9828/3276 paths, scan interval 60 secs Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd 198.51.101.1 4 65000 90518 90507 14743 0 0 8w0d 1638 198.51.102.2 4 65000 4901 4895 14743 0 0 2d01h 1638 198.51.103.3 4 65000 4903 4895 14743 0 0 2d01h 1638	

VPLS BGP Signaling L2VPN Inter-AS Option A: Example

The following example configuration describes Inter-AS Option A for VPLS BGP signaling in an L2VPN. The Autonomous System Boundary Router (ASBR) 1 acts as the Provider Edge (PE) for all VPLS instances that span over Autonomous System (AS) 1 and ASBR 2 are viewed as the CE device. And for the other way around, for AS 2, ASBR 2 acts as the PE and ASBR 1 is viewed as the CE. MPLS is not required between ASBR 1 and ASBR 2 because VPLS is used for layer 2 linking. Each VPLS instance needs to be segregated so that it can be sent in the proper VPLS domain in ASBRs (for example, a switchport interface or Ethernet sub-interface).

Note From a BGP signaling perspective, there is no specific change within the AS. From the VPLS perspective, there is no BGP peering between ASBR1 and ASBR2.

The following figure shows a network diagram for the BGP signaling Inter-AS option A BGP configuration:



The following example shows the PE 1 BGP configuration for Inter-AS Option A:

```
router bgp 100
  neighbor 10.0.0.2 remote-as 100
   address-family l2vpn vpls
    neighbor 10.0.0.2 activate
    neighbor 10.0.0.2 send-community extended
    neighbor 10.0.0.2 suppress-signaling-protocol ldp
   exit-address-family
```

The following example shows the ASBR 1 BGP configuration for Inter-AS Option A:

```
router bgp 100
  neighbor 10.0.0.1 remote-as 100
   address-family l2vpn vpls
    neighbor 10.0.0.1 activate
    neighbor 10.0.0.1 send-community extended
```

```
    neighbor 10.0.0.1 suppress-signaling-protocol ldp
   exit-address-family
```

The following example shows the ASBR 2 BGP configuration for Inter-AS Option A:

```
router bgp 200
  neighbor 10.0.1.1 remote-as 100
   address-family l2vpn vpls
    neighbor 10.0.1.1 activate
    neighbor 10.0.1.1 send-community extended
    neighbor 10.0.1.1 suppress-signaling-protocol ldp
   exit-address-family
```

The following example shows the PE 2 BGP configuration for Inter-AS Option A:

```
router bgp 200
  neighbor 10.0.1.2 remote-as 100
   address-family l2vpn vpls
    neighbor 10.0.1.2 activate
    neighbor 10.0.1.2 send-community extended
    neighbor 10.0.1.2 suppress-signaling-protocol ldp
   exit-address-family
```

Additional References for VPLS Autodiscovery BGP Based

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference

Standards and RFCs

Standard/RFC	Title
draft-ietf-l2vpn-signaling-08.txt	*Provisioning, Autodiscovery, and Signaling in L2VPNs*
draft-ietf-l2vpn-vpls-bgp-08.8	*Virtual Private LAN Service (VPLS) Using BGP for Autodiscovery and Signaling*
draft-ietf-mpls-lsp-ping-03.txt	*Detecting MPLS Data Plane Failures*
draft-ietf-pwe3-vccv-01.txt	*Pseudo-Wire (PW) Virtual Circuit Connection Verification (VCCV)*
RFC 3916	*Requirements for Pseudo-wire Emulation Edge-to-Edge (PWE3)*
RFC 3981	*Pseudo Wire Emulation Edge-to-Edge Architecture*
RFC 6074	Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)
RFC 4761	Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling

MIBs

MIB	MIBs Link
• CISCO-IETF-PW-ATM-MIB (PW-ATM-MIB) • CISCO-IETF-PW-ENET-MIB (PW-ENET-MIB) • CISCO-IETF-PW-FR-MIB (PW-FR-MIB) • CISCO-IETF-PW-MIB (PW-MIB) • CISCO-IETF-PW-MPLS-MIB (PW-MPLS-MIB)	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register on Cisco.com.	http://www.cisco.com/techsupport

Feature Information for VPLS BGP Signaling L2VPN Inter-AS Option A

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 39: Feature Information for VPLS BGP Signaling L2VPN

Feature Name	Releases	Feature Information
VPLS BGP Signaling L2VPN	Cisco IOS XE Release 3.8S	This feature simplifies the auto-discovery and signaling of all known provider edge (PE) devices in a VPLS instance by using BGP for both functions. The following commands were introduced or modified: **autodiscovery bgp signaling bgp**, **debug bgp l2vpn vpls updates**, **neighbor suppress-signaling-protocol ldp**, **ve id**, **ve range**, **show bgp l2vpn vpls**.

CHAPTER 26

VPLS BGP Signaling L2VPN Inter-AS Option B

The VPLS BGP Signaling L2VPN Inter-AS Option B feature simplifies the auto-discovery and signaling of all known provider edge (PE) devices in a Virtual Private LAN Switching (VPLS) instance by using Border Gateway Protocol (BGP). This document describes how to configure the VPLS BGP Signaling L2VPN Inter-AS Option B feature.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for VPLS BGP Signaling L2VPN Inter-AS Option B

- Disable control word for Virtual Private LAN Switching (VPLS) Border Gateway Protocol (BGP) signaling by using the **no control-word** command under a pseudowire class. For example:

```
Device> enable
Device# configure terminal
Device(config)# pseudowire-class my-pw-class
Device(config-pw-class)# no control-word
```

- The route distinguisher (RD) must match for all the virtual forwarding instances (VFIs) in a VPLS domain.
- Ensure that the L2VPN VPLS Inter-AS Option B feature is configured on Autonomous System Boundary Routers (ASBRs) and PE devices.

Information About VPLS BGP Signaling L2VPN Inter-AS Option B

BGP Auto-discovery and Signaling for VPLS

The Virtual Private LAN Switching (VPLS) control plane is used for auto-discovery and signaling. Auto-discovery involves locating all provider edge (PE) devices that participate in a particular VPLS instance. Signaling is accomplished by configuring pseudowires for a VPLS instance. Prior to the introduction of the VPLS BGP Signaling L2VPN Inter-AS Option B feature, Label Distribution Protocol (LDP) was used for signaling and Border Gateway Protocol (BGP) was used for auto-discovery, as specified in RFC 6074. With the introduction of the VPLS BGP Signaling L2VPN Inter-AS Option B feature, the VPLS BGP Signaling L2VPN feature supports RFC 4761 by simplifying the auto-discovery and signaling of all known PE devices in a VPLS instance by using BGP for both functions. Auto-discovery is defined per VPLS instance.

Internal BGP (IBGP) peers exchange update messages of the L2VPN Address Family Identifier (AFI) and the Subsequent Address Family Identifier (SAFI) numbers with L2VPN information to perform both auto-discovery and signaling, which includes the Network Layer Reachability Information (NLRI).

Both BGP standards (RFC 6074 and RFC 4761) for the auto-discovery protocol for VPLS use the same BGP AFI (25) and SAFI (65) but they have different Network Layer Reachability Information (NLRI) encoding, which makes them incompatible with each other. CLI configuration is needed to distinguish the two encoding types as they are mutually exclusive per neighbor. The difference between the two BGP standards is:

- RFC 6074 provides guidelines for specifying length encoding as bits.
- RFC 4761 provides guidelines for specifying length encoding as bytes.

To detect which NLRI encoding standard is supported, the length encoding needs to be determined.

BGP L2VPN Signaling with NLRI

Network Layer Reachability Information (NLRI) enables Border Gateway Protocol (BGP) to carry supernetting information, as well as perform aggregation. Each NLRI consists of block labels that follow the structure LB, LB+1, ... , LB+VBS-1. The NLRI is exchanged between BGP devices for BGP auto-discovery with BGP signaling. The following fields are configured or auto-generated for each Virtual Private LAN Switching (VPLS) instance:

- Length (2 Octets)
- Route distinguisher (RD) is usually an auto-generated 8-byte VPN ID that can also be configured. This value must be unique for a VPLS bridge-domain (or instance).
- VPLS Endpoint ID (VEID) (2 Octets). Each PE device is configured with a VEID value.

- VPLS Endpoint Block Offset (VBO) (2 Octets).
- VPLS Endpoint Block Size (VBS) (2 Octets).
- Label Base (LB) (3 Octets).
- Extended Community Type (2 Octets) - 0x800A attributes. The Route Target (RT) specified for a VPLS instance, next-hop and other Layer 2 information is carried in this encoding. An RT-based import and export mechanism similar to L3VPN is performed by BGP to perform filtering on the L2VPN NLRIs of a particular VPLS instance.
- Encapsulation Type (1 Octet) - VPLS = 19
- Control Flags (1 Octet)
- Layer 2 Maximum Transmission Unit (MTU) (2 Octets)
- Reserved (2 Octets)

How to Configure VPLS BGP Signaling L2VPN Inter-AS Option B

Enabling BGP Auto-discovery and BGP Signaling

Perform this task to enable Virtual Private LAN Service (VPLS) PE devices to discover other PE devices by BGP auto-discovery and BGP signaling functions announced through IBGP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-context-name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling bgp**
6. **ve id** *ve-ID-number*
7. **ve range** *ve-range-number*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**l2vpn vfi context** *vfi-context-name* **Example:** `Device(config)# l2vpn vfi context vfi1`	Establishes a Layer 2 VPN (L2VPN) virtual forwarding interface (VFI) for specifying core-facing pseudowires in a Virtual Private LAN Services (VPLS) and enters L2VFI configuration mode. • The VFI represents an emulated LAN or a VPLS forwarder from the VPLS architectural model when using an emulated LAN interface.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling bgp** **Example:** `Device(config-vfi)# autodiscovery bgp signaling bgp`	Enables BGP auto-discovery and BGP signaling on the device.
Step 6	**ve id** *ve-ID-number* **Example:** `Device(config-vfi)# ve id 1`	Configures a VPLS Endpoint ID (VEID) for the NLRI exchanged between BGP devices for BGP auto-discovery with BGP signaling. • For example, VEID numbering sequences such as 1,2,3 or 501, 502, 503 are preferred because the VEIDs are contiguous. • Avoid a non-contiguous numbering scheme such as 100, 200, 300. Repeat this step to add more VEIDs. The VEID must be unique within the same VPLS domain for all PE devices. **Note** If you change the VEID, then the virtual circuit (VC) reprovisions and traffic is impacted as a result.
Step 7	**ve range** *ve-range-number* **Example:** `Device(config-vfi)# ve range 10`	Overrides the minimum size of VPLS edge (VE) blocks. • The VE range value should be approximately the same as the number of neighbors (up to 100). • The VE range can be configured based on the number of neighboring PE devices in the network. • For example, if 50 PE devices are in a VPLS domain, then a VE range of 50 is better than 10 because the number of NLRIs exchanged are less and the convergence time is reduced.

	Command or Action	Purpose
		Note If no VE range is configured or an existing VE range value is removed, then the default VE range of 10 is applied. The default VE range should not be used if the device has many PE neighbors. **Note** If you change the VE range, then the VC reprovisions and traffic is impacted as a result.
Step 8	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. **Note** Commands take effect after the device exits L2VFI configuration mode.

Configuring BGP Signaling for VPLS Autodiscovery

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp graceful-restart**
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
6. **address-family l2vpn vpls**
7. **neighbor** {*ip-address* | *peer-group-name*} **activate**
8. **neighbor** {*ip-address* | *peer-group-name*} **send-community extended**
9. **neighbor** {*ip-address* | *peer-group-name*} **suppress-signaling-protocol ldp**
10. **exit-address-family**
11. Repeat steps 1 to 10 to configure and activate other BGP neighbors in an L2VPN address family.
12. **end**
13. **show l2vpn vfi**
14. **show ip bgp l2vpn vpls** {**all** [**summary**] | **rd** *route-distinguisher*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.
Step 4	**bgp graceful-restart** **Example:** `Device(config-router)# bgp graceful-restart`	Enables the Border Gateway Protocol (BGP) graceful restart capability globally for all BGP neighbors.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 198.51.100.1 remote-as 65000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor at 10.10.10.1 is an internal BGP neighbor.
Step 6	**address-family l2vpn vpls** **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The **vpls** keyword specifies that the VPLS endpoint provisioning information is to be distributed to BGP peers and a L2VPN VPLS address family session is created.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 198.51.100.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **send-community extended** **Example:** `Device(config-router-af)# neighbor 198.51.100.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **suppress-signaling-protocol ldp** **Example:** `Device(config-router-af)# neighbor 198.51.100.1 suppress-signaling protocol ldp`	Suppresses LDP signaling for a BGP neighbor so that BGP signaling for VPLS auto-discovery is used instead. • In this example, LDP signaling is suppressed for the neighbor at 10.10.10.1.

	Command or Action	Purpose
Step 10	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.
Step 11	Repeat steps 1 to 10 to configure and activate other BGP neighbors in an L2VPN address family.	
Step 12	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

Step 13	**show l2vpn vfi**
	Example:
	`Device# show l2vpn vfi`
	<pre>PE1-standby#sh l2vpn vfi
	Load for five secs: 0%/0%; one minute: 0%; five
	minutes: 0%
	Time source is hardware calendar, *20:50:52.526
	GMT Wed Aug 29 2012
	
	Legend: RT=Route-target, S=Split-horizon, Y=Yes,
	 N=No
	
	VFI name: VFI1, state: up, type: multipoint,
	signaling: BGP
	  VPN ID: 1, VE-ID: 10, VE-SIZE: 10
	  RD: 1:1, RT: 1:1
	  Bridge-Domain 100 attachment circuits:
	  Pseudo-port interface: pseudowire100001
	  Interface Peer Address VE-ID Local
	 Label Remote Label S
	  pseudowire100003 198.51.100.2 11 1003
	         2002 Y
	  pseudowire100005 198.51.100.3 12 1004
	         2002 Y
	
	VFI name: VFI2, state: up, type: multipoint,
	signaling: BGP
	  VPN ID: 2, VE-ID: 20, VE-SIZE: 12
	  RD: 1:2, RT: 1:2, import 3:3, export 4:4
	  Bridge-Domain 200 attachment circuits:
	  Pseudo-port interface: pseudowire100002
	  Interface Peer Address VE-ID Local
	 Label Remote Label S
	  pseudowire100004 198.51.100.2 21
	 1021 2020 Y
	  pseudowire100006 198.51.100.3 22
	 1022 2020 Y</pre>
		Displays information about the configured VFI instances.

Step 14	**show ip bgp l2vpn vpls** {**all** [**summary**] | **rd** *route-distinguisher*} **Example:** `Device# show ip bgp l2vpn vpls all summary`	Displays information about the L2VPN VPLS address family.

	Command or Action	Purpose

	`BGP router identifier 198.51.100.1, local AS number 65000`
	`BGP table version is 14743, main routing table version 14743`
	`6552 network entries using 1677312 bytes of memory`
	`6552 path entries using 838656 bytes of memory`
	`3276/3276 BGP path/bestpath attribute entries using 760032 bytes of memory`
	`1638 BGP extended community entries using 65520 bytes of memory`
	`0 BGP route-map cache entries using 0 bytes of memory`
	`0 BGP filter-list cache entries using 0 bytes of memory`
	`BGP using 3341520 total bytes of memory`
	`BGP activity 9828/3276 prefixes, 9828/3276 paths, scan interval 60 secs`
	
	`Neighbor        V           AS MsgRcvd MsgSent TblVer  InQ OutQ Up/Down  State/PfxRcd`
	`198.51.101.1     4        65000   90518   90507   14743    0    0 8w0d         1638`
	`198.51.102.2     4        65000    4901    4895   14743    0    0 2d01h        1638`
	`198.51.103.3     4        65000    4903    4895   14743    0    0 2d01h        1638`	

Configuration Examples for L2VPN VPLS Inter-AS Option B

Example: VPLS BGP Signaling L2VPN Inter-AS Option B

The following example configuration describes Inter-AS Option B for VPLS BGP signaling in a Layer 2 VPN. BGP MPLS forwarding is required between ASBR 1 and ASBR 2.

Note From a BGP signaling perspective, there is no specific change within the autonomous system. From the VPLS perspective, there is EBGP peering between ASBR1 and ASBR2.

The following figure shows a network diagram for the BGP signaling Inter-AS option B BGP configuration:

Figure 44: VPLS BGP Signaling L2VPN Inter-AS Option B Sample Topology

The following example shows the PE 1 BGP configuration for Inter-AS Option B:

```
l2vpn vfi context TEST101
 vpn id 1
 autodiscovery bgp signaling bgp
```

```
  ve id 1
  route-target import 22:22
  route-target export 11:11
  no auto-route-target
!
mpls ldp graceful-restart
!
bridge-domain 1
 member GigabitEthernet0/0/7 service-instance 101
 member vfi TEST101
!
interface Loopback0
 ip address 198.51.101.2 255.255.255.255
!
interface GigabitEthernet0/0/1
 description - connects to RR1
 ip address 200.1.1.1 255.255.255.0
 negotiation auto
 mpls ip
!
interface GigabitEthernet0/0/7
 description - connects to CE1
 no ip address
 negotiation auto
 service instance 101 ethernet
  encapsulation dot1q 101
  rewrite ingress tag pop 1 symmetric
 !
!
router ospf 10
 nsf
 network 200.1.1.0 0.0.0.255 area 0
 network 198.51.101.2 0.0.0.0 area 0
!
router bgp 10
 bgp log-neighbor-changes
 bgp update-delay 1
 bgp graceful-restart restart-time 120
 bgp graceful-restart stalepath-time 360
 bgp graceful-restart
 no bgp default ipv4-unicast
 neighbor 200.1.1.1 remote-as 10
 neighbor 200.1.1.1 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 200.1.1.1 activate
  neighbor 200.1.1.1 send-community extended
  neighbor 200.1.1.1 suppress-signaling-protocol ldp
 exit-address-family
!
```

The following example shows the ASBR 1 BGP configuration for Inter-AS Option B:

```
router bgp 10
bgp log-neighbor-changes
bgp update-delay 1
bgp graceful-restart restart-time 120
bgp graceful-restart stalepath-time 360
bgp graceful-restart
no bgp default ipv4-unicast
no bgp default route-target filter
 neighbor 192.0.2.1 remote-as 10
```

```
 neighbor 192.0.2.1 update-source Loopback0
 neighbor 203.0.203.1 remote-as 20
 neighbor 203.0.203.1 ebgp-multihop 255
 neighbor 203.0.203.1 update-source Loopback0
!
address-family ipv4
 exit-address-family
!
address-family l2vpn vpls
 neighbor 192.0.2.1 activate
 neighbor 192.0.2.1 send-community extended
 neighbor 192.0.2.1 next-hop-self
 neighbor 192.0.2.1 suppress-signaling-protocol ldp
 neighbor 203.0.203.1 activate
 neighbor 203.0.203.1 send-community extended
 neighbor 203.0.203.1 next-hop-self
 neighbor 203.0.203.1 suppress-signaling-protocol ldp
 exit-address-family
```

The following example shows the ASBR 2 BGP configuration for Inter-AS Option B:

```
mpls ldp graceful-restart
!
interface Loopback0
 ip address 203.0.203.1 255.255.255.255
!
interface GigabitEthernet0/0/1
 description - connects to RR1
 ip address 192.0.2.2 255.255.255.0
 negotiation auto
 mpls ip
 mpls bgp forwarding
!
interface GigabitEthernet0/2/1
 description - connects to ASBR3
 ip address 192.0.2.200 255.255.255.0
 negotiation auto
 mpls ip
 mpls bgp forwarding
!
router ospf 10
 nsf
 network 192.0.2.0 0.0.0.255 area 0
 network 203.0.203.1 0.0.0.0 area 0
 network 0.0.0.0 255.255.255.255 area 0
!
router bgp 10
 bgp log-neighbor-changes
 bgp update-delay 1
 bgp graceful-restart restart-time 120
 bgp graceful-restart stalepath-time 360
 bgp graceful-restart
 no bgp default ipv4-unicast
 no bgp default route-target filter
 neighbor 203.0.203.3 remote-as 20
 neighbor 203.0.203.3 ebgp-multihop 255
 neighbor 203.0.203.3 update-source Loopback0
 neighbor 203.0.203.2 remote-as 10
 neighbor 203.0.203.2 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 203.0.203.3 activate
```

```
  neighbor 203.0.203.3 send-community extended
  neighbor 203.0.203.3 next-hop-self
  neighbor 203.0.203.3 suppress-signaling-protocol ldp
  neighbor 203.0.203.2 activate
  neighbor 203.0.203.2 send-community extended
  neighbor 203.0.203.2 next-hop-self
  neighbor 203.0.203.2 suppress-signaling-protocol ldp
 exit-address-family
```

The following example shows the PE 2 BGP configuration for Inter-AS Option B:

```
l2vpn vfi context TEST101
 vpn id 1
 autodiscovery bgp signaling bgp
  ve id 2
  route-target import 22:22
  route-target export 11:11
  no auto-route-target
!
mpls ldp graceful-restart
!
bridge-domain 1
 member GigabitEthernet0/0/7 service-instance 101
 member vfi TEST101
!
interface Loopback0
 ip address 192.0.2.3 255.255.255.255
!
interface GigabitEthernet0/0/1
 description - connects to RR1
 ip address 192.0.2.1 255.255.255.0
 negotiation auto
 mpls ip
!
interface GigabitEthernet0/0/7
 description - connects to CE2
 no ip address
 negotiation auto
 service instance 101 ethernet
  encapsulation dot1q 101
  rewrite ingress tag pop 1 symmetric
 !
!
router ospf 10
 nsf
 network 192.0.2.0 0.0.0.255 area 0
 network 192.0.2.3 0.0.0.0 area 0
!
router bgp 10
 bgp log-neighbor-changes
 bgp update-delay 1
 bgp graceful-restart restart-time 120
 bgp graceful-restart stalepath-time 360
 bgp graceful-restart
 no bgp default ipv4-unicast
 neighbor 211.1.1.1 remote-as 10
 neighbor 211.1.1.1 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 211.1.1.1 activate
  neighbor 211.1.1.1 send-community extended
```

```
  neighbor 211.1.1.1 suppress-signaling-protocol ldp
 exit-address-family
```

The following example shows the route reflector device BGP configuration for Inter-AS Option B:

```
mpls ldp graceful-restart
!
interface Loopback0
 ip address 203.0.203.1 255.255.255.255
!
interface GigabitEthernet1/1
 description - connects to PE1
 ip address 203.0.203.2 255.255.255.0
 mpls ip
!
interface GigabitEthernet1/2
 description - connects to PE2
 ip address 203.0.203.3 255.255.255.0
 mpls ip
!
interface GigabitEthernet1/5
 description - connects to ASBR1
 ip address 203.0.203.4 255.255.255.0
 mpls ip
 mpls bgp forwarding
!
interface GigabitEthernet1/6
 description - connects to ASBR2
 ip address 203.0.203.5 255.255.255.0
 mpls ip
 mpls bgp forwarding
!
router ospf 10
 nsf
 network 203.0.203.6 0.0.0.255 area 0
 network 203.0.203.7 0.0.0.255 area 0
 network 203.0.203.8 0.0.0.255 area 0
 network 203.0.203.9 0.0.0.255 area 0
 network 203.0.203.1 0.0.0.0 area 0
!
router bgp 10
 bgp log-neighbor-changes
 bgp update-delay 1
 bgp graceful-restart restart-time 120
 bgp graceful-restart stalepath-time 360
 bgp graceful-restart
 no bgp default ipv4-unicast
 neighbor 203.0.203.11 remote-as 10
 neighbor 203.0.203.11 update-source Loopback0
 neighbor 203.0.203.12 remote-as 10
 neighbor 203.0.203.12 update-source Loopback0
 neighbor 203.0.203.13 remote-as 10
 neighbor 203.0.203.13 update-source Loopback0
 neighbor 203.0.203.14 remote-as 10
 neighbor 203.0.203.14 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family l2vpn vpls
  neighbor 203.0.203.11 activate
  neighbor 203.0.203.11 send-community extended
  neighbor 203.0.203.11 route-reflector-client
  neighbor 203.0.203.11 suppress-signaling-protocol ldp
  neighbor 203.0.203.12 activate
```

```
  neighbor 203.0.203.12 send-community extended
  neighbor 203.0.203.12 route-reflector-client
  neighbor 203.0.203.12 suppress-signaling-protocol ldp
  neighbor 203.0.203.13 activate
  neighbor 203.0.203.13 send-community extended
  neighbor 203.0.203.13 route-reflector-client
  neighbor 203.0.203.13 suppress-signaling-protocol ldp
  neighbor 203.0.203.14 activate
  neighbor 203.0.203.14 send-community extended
  neighbor 203.0.203.14 route-reflector-client
  neighbor 203.0.203.14 suppress-signaling-protocol ldp
 exit-address-family
!
```

Additional References for VPLS BGP Signaling L2VPN Inter-AS Option B

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference
L2VPN VPLS Inter-AS Option B	*L2VPN VPLS Inter-AS Option B*
VPLS Autodiscovery: BGP Based	*VPLS Autodiscovery BGP Based*
VPLS BGP Signaling L2VPN Inter-AS Option A	*VPLS BGP Signaling L2VPN Inter-AS Option A*

Standards and RFCs

Standard and RFC	Title
draft-kothari-l2vpn-auto-site-id-01.txt	*Automatic Generation of Site IDs for Virtual Private LAN Service*
draft-ietf-l2vpn-vpls-multihoming-03.txt	*BGP based Multi-homing in Virtual Private LAN Service*
RFC 6074	*Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)*
RFC 4761	*Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*

MIBs

MIB	MIBs Link
• CISCO-IETF-PW-ATM-MIB (PW-ATM-MIB) • CISCO-IETF-PW-ENET-MIB (PW-ENET-MIB) • CISCO-IETF-PW-FR-MIB (PW-FR-MIB) • CISCO-IETF-PW-MIB (PW-MIB) • CISCO-IETF-PW-MPLS-MIB (PW-MPLS-MIB)	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for VPLS BGP Signaling L2VPN Inter-AS Option B

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 40: Feature Information for VPLS BGP Signaling L2VPN Inter-AS Option B

Feature Name	Releases	Feature Information
VPLS BGP Signaling L2VPN Inter-AS Option B	Cisco IOS XE Release 3.12S	This feature simplifies the auto-discovery and signaling of all known provider edge (PE) devices in a VPLS instance by using BGP for both functions. The following command was modified: **show mpls forwarding**

CHAPTER 27

Frame Relay over L2TPv3

The Frame Relay over L2TPv3 (FRoL2TPv3) feature enables Frame Relay switching over Layer 2 Tunnel Protocol Version 3 (L2TPv3). The feature works with like interfaces and disparate interfaces (L2VPN interworking).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring Frame Relay over L2TPv3

Before configuring Frame Relay over L2TPv3, you should understand how to configure Layer 2 VPNs and Frame Relay. See the “Additional References” section in this chapter for pointers to the feature modules that explain how to configure and use Layer 2 VPNs and Frame Relay.

Restrictions for Configuring Frame Relay over L2TPv3

The following functionalities are not supported:

- Frame Relay to 802.1Q/QinQ VLAN interworking

- Frame Relay-to-Ethernet routed interworking
- Frame Relay port-to-port switching
- L2TPv3 pseudowire redundancy for Frame Relay

Information About Configuring Frame Relay over L2TPv3

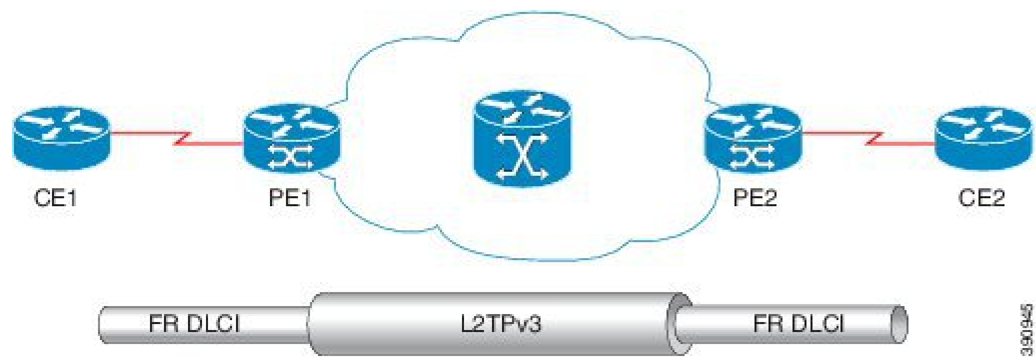
Frame Relay over L2TPv3 Overview

Frame Relay over L2TPv3 enables provider edge (PE) devices to forward Frame Relay frames to pseudowires based on the receiving interface and the Data-Link Connection Identifier (DLCI) number. PE devices also provide Local Management Interface (LMI)-based signaling to customer edge (CE) devices, emulating Frame Relay switches.

In Frame Relay over L2TPv3, the Frame Relay header is retained at the ingress PE device. The device does not reconstruct the Frame Relay header before forwarding packets to the CE device.

The figure below shows a Frame Relay over L2TPv3 topology.

Figure 45: Frame Relay over L2TPv3



Frame Relay over L2TPv3 supports the following functionalities:

- Frame Relay data-link connection identifier (DLCI)-to-Frame Relay DLCI
- Frame Relay DLCI-to-Ethernet port / 802.1Q / QinQ bridged interworking
- Local Management Interface (LMI)
- L2TPv3 sequencing
- L2TPv3 tunnel marking

How to Configure Frame Relay over L2TPv3

Configuring Frame Relay over L2TPv3 without LMI

This section explains how to configure Frame Relay over L2TPv3 without enabling Local Management Interface (LMI).

On CE1

The CE1 device receives the Frame Relay frames forwarded by the PE1 device over the Frame Relay link. On CE1, configure an interface and a DLCI number based on which the PE1 device forwards traffic to the appropriate pseudowire.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **encapsulation frame-relay** [**cisco** | **ietf**]
6. **no keepalive**
7. **frame-relay intf-type dce**
8. **exit**
9. **interface** *type number* **point-to-point**
10. **ip address** *ip-address mask*
11. **frame-relay interface-dlci** *dlci*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface serial3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**encapsulation frame-relay** [**cisco** | **ietf**] **Example:**	Specifies Frame Relay encapsulation for the interface. • You can specify different types of encapsulations. • You can set one interface to Cisco encapsulation and the other interface to IETF encapsulation.

	Command or Action	Purpose
	`Device(config-if)# encapsulation frame-relay ietf`	
Step 6	**no keepalive** **Example:** `Device(config-if)# no keepalive`	Disables the keepalive configuration.
Step 7	**frame-relay intf-type dce** **Example:** `Device(config-if)# frame-relay intf-type dce`	Specifies that the interface is a DCE switch. • You can also specify the interface to support Network-to-Network Interface (NNI) and DTE connections.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **point-to-point** **Example:** `Device(config)# interface serial 3/1/0.1 point-to-point`	Specifies a serial interface and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 198.51.100.2 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 11	**frame-relay interface-dlci** *dlci* **Example:** `Device(config-if)# frame-relay interface-dlci 25`	Assigns a data-link connection identifier (DLCI) to the Frame Relay interface.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode. After configuring CE1, you can configure CE2 in a similar manner.

On PE1

The PE1 device forwards Frame Relay frames to the appropriate pseudowire, based on the receiving interface and DLCI number configured on the CE1 device.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **interface** *type number*
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **encapsulation frame-relay** [**cisco** | **ietf**]
6. **no keepalive**
7. **pseudowire-class** [*pw-class-name*]
8. **encapsulation l2tpv3**
9. **ip local interface loopback** *loopback id*
10. **connect** *connection-name interface dlci* **l2transport**
11. **xconnect** *peer-router-id vcid* **encapsulation l2tpv3 pw-class l2tpv3**
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface serial3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**encapsulation frame-relay** [**cisco** | **ietf**] **Example:** `Device(config-if)# encapsulation frame-relay ietf`	Specifies Frame Relay encapsulation for the interface. • You can specify different types of encapsulations. • You can set one interface to Cisco encapsulation and the other interface to IETF encapsulation.
Step 6	**no keepalive** **Example:** `Device(config-if)# no keepalive`	Disables the keepalive configuration.

	Command or Action	Purpose
Step 7	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class l2tpv3`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 8	**encapsulation l2tpv3** **Example:** `Device(config-pw)# encapsulation l2tpv3`	Specifies the tunneling encapsulation as L2TPv3.
Step 9	**ip local interface loopback** *loopback id* **Example:** `Device(config-pw)# ip local interface Loopback0`	Specifies the local loopback interface on PE1 for the L2TPv3 tunnel.
Step 10	**connect** *connection-name interface dlci* **l2transport** **Example:** `Device(config)# connect fr1 serial5/0 1000 l2transport`	Defines connections between Frame Relay Permanent Virtual Circuits (PVCs) and enters connect configuration mode. • Using the **l2transport** keyword specifies that the PVC is not a locally switched PVC, but is tunneled over the backbone network. • The *connection-name* argument is a text string that you provide. • The *interface* argument is the interface on which a PVC connection is defined. • The *dlci* argument is the DLCI number of the PVC that is connected.
Step 11	**xconnect** *peer-router-id vcid* **encapsulation l2tpv3 pw-class l2tpv3** **Example:** `Device(config-xconnect-conn-config)# xconnect 198.51.100.2 123 encapsulation l2tpv3 pw-class l2tpv3`	Creates the VC to transport the Layer 2 packets. • In a DLCI-to DLCI connection type, Frame Relay over L2TPv3 uses the **xconnect** command in connect configuration mode. • The *vcid* or identifier of the virtual circuit (VC) between the PE devices should be the same on both devices that are being connected.
Step 12	**end** **Example:** `Device(config-xconnect-conn-config)# end`	Exits connect configuration mode and returns to privileged EXEC mode. After configuring PE1, you can configure PE2 in a similar manner.

Configuring Frame Relay over L2TPv3 with LMI

This section explains how to configure Frame Relay over L2TPv3 with Local Management Interface (LMI) enabled.

On CE1

The CE1 device receives the Frame Relay frames forwarded by the PE1 device over the Frame Relay link. On CE1, configure an interface and a DLCI number based on which the PE1 device forwards traffic to the appropriate pseudowire. Local Management Interface (LMI) is also tunneled over the pseudowire. Therefore, you need to properly configure the customer edge (CE) device for LMI.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot/subslot /port* [**.** *subinterface*]
4. **no ip address** [*ip-address mask*] [**secondary**]
5. **encapsulation frame-relay** [**cisco** | **ietf**]
6. **frame-relay intf-type dce**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *slot/subslot /port* [**.** *subinterface*] **Example:** `Device(config)# interface serial3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 4	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**encapsulation frame-relay** [**cisco**	**ietf**] **Example:** `Device(config-if)# encapsulation frame-relay ietf`	Specifies Frame Relay encapsulation for the interface. • You can specify different types of encapsulations. • You can set one interface to Cisco encapsulation and the other interface to IETF encapsulation.
Step 6	**frame-relay intf-type dce** **Example:**	Specifies that the interface is a Data Communications Equipment (DCE) switch.

	Command or Action	Purpose
	`Device(config-if)# frame-relay intf-type dce`	• You can also specify the interface to support Network-to-Network Interface (NNI) and Data Transmission Equipment (DTE) connections.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode. After configuring CE1, you can configure CE2 in a similar manner.

On PE1

The PE1 device forwards Frame Relay frames to the CE1 device over the Frame Relay link. The PE1 device also provides Local Management Interface (LMI) signaling to the CE1 device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot/subslot/port* [**.** *subinterface*]
4. **encapsulation frame-relay** [**cisco** | **ietf**]
5. **pseudowire-class** [*pw-class-name*]
6. **encapsulation l2tpv3**
7. **ip local interface loopback** *loopback id*
8. **connect** *connection-name interface dlci* **l2transport**
9. **xconnect** *peer-router-id vcid* **encapsulation l2tpv3 pw-class l2tpv3**
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *slot/subslot/port* [**.** *subinterface*] **Example:** `Device(config)# interface serial3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 4	**encapsulation frame-relay** [**cisco** | **ietf**]	Specifies Frame Relay encapsulation for the interface.

	Command or Action	Purpose
	Example: `Device(config-if)# encapsulation frame-relay ietf`	• You can specify different types of encapsulations. • You can set one interface to Cisco encapsulation and the other interface to IETF encapsulation.
Step 5	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class l2tpv3`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 6	**encapsulation l2tpv3** **Example:** `Device(config-pw)# encapsulation l2tpv3`	Specifies the tunneling encapsulation as L2TPv3.
Step 7	**ip local interface loopback** *loopback id* **Example:** `Device(config-pw)# ip local interface Loopback0`	Specifies the local loopback interface.
Step 8	**connect** *connection-name interface dlci* **l2transport** **Example:** `Device(config)# connect fr1 serial5/0 1000 l2transport`	Defines connections between Frame Relay Permanent Virtual Circuits (PVCs) and enters connect configuration mode. • Using the **l2transport** keyword specifies that the PVC is not a locally switched PVC, but is tunneled over the backbone network. • The *connection-name* argument is a text string that you provide. • The *interface* argument is the interface on which a PVC connection is defined. • The *dlci* argument is the DLCI number of the PVC that is connected.
Step 9	**xconnect** *peer-router-id vcid* **encapsulation l2tpv3 pw-class l2tpv3** **Example:** `Device(config-fr-pw-switching)# xconnect 198.51.100.2 123 encapsulation l2tpv3 pw-class l2tpv3`	Creates the virtual circuit (VC) to transport the Layer 2 packets. • In a DLCI-to-DLCI connection type, Frame Relay over L2TPv3 uses the **xconnect** command in connect configuration mode.
Step 10	**end** **Example:** `Device(config-fr-pw-switching)# end`	Exits connect configuration mode and returns to privileged EXEC mode. After configuring PE1, you can configure PE2 in a similar manner.

Configuring Frame Relay L2TPv3 Tunnel Marking

L2TPv3 Tunnel Marking introduces the capability to define and control the quality of service (QoS) for incoming customer traffic on the provider edge (PE) device in a service provider network.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-name*
4. **match fr-dlci** *dlci-number*
5. **policy-map dlci** *dlci-number*
6. **class** *class-name*
7. **set ip precedence tunnel** *precedence-value*
8. **interface serial** *slot*/*subslot*/*port* [**.** *subinterface*]
9. **no ip address** [*ip-address mask*] [**secondary**]
10. **encapsulation frame-relay** [**cisco** | **ietf**]
11. **no keepalive**
12. **service-policy input** *policy-name*
13. **end**
14. **pseudowire-class** [*pw-class-name*]
15. **encapsulation l2tpv3**
16. **ip local interface loopback** *loopback id*
17. **connect** *connection-name interface dlci* **l2transport**
18. **xconnect** *peer-router-id vcid* **encapsulation l2tpv3 pw-class l2tpv3**
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map** *class-name* **Example:** `Device(config)# class-map class1`	Specifies the user-defined name of the traffic class and enters class map configuration mode.

	Command or Action	Purpose
Step 4	**match fr-dlci** *dlci-number* **Example:** `Device(config-cmap)# match fr-dlci 50`	Specifies the number of the Data-Link Connection Identifier (DLCI) associated with the packet as a match criterion in the class map.
Step 5	**policy-map dlci** *dlci-number* **Example:** `Device(config-cmap)# policy-map dlci 50`	Specifies the type of policy map as DLCI and enters policy map configuration mode.
Step 6	**class** *class-name* **Example:** `Device(config-pmap)# class class1`	Specifies the name of a predefined traffic class, which was configured with the **class-map** command, used to classify traffic to the traffic policy and enters policy-map class configuration mode.
Step 7	**set ip precedence tunnel** *precedence-value* **Example:** `Device(config-pmap-c)# set ip precedence tunnel 2`	Sets the precedence value in the header of the L2TPv3 tunneled packet for tunnel marking.
Step 8	**interface serial** *slot/subslot/port* [**.** *subinterface*] **Example:** `Device(config-pmap-c)# interface serial3/1/0`	Specifies a serial interface and enters interface configuration mode.
Step 9	**no ip address** [*ip-address mask*] [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 10	**encapsulation frame-relay** [**cisco** | **ietf**] **Example:** `Device(config-if)# encapsulation frame-relay ietf`	Specifies Frame Relay encapsulation for the interface. • You can specify different types of encapsulations. • You can set one interface to Cisco encapsulation and the other interface to IETF encapsulation.
Step 11	**no keepalive** **Example:** `Device(config-if)# no keepalive`	Disables the keepalive configuration.
Step 12	**service-policy input** *policy-name* **Example:** `Device(config-if)# service-policy input policy1`	Attaches a traffic policy to the interface.

	Command or Action	Purpose
Step 13	**end** **Example:** `Device(config-if)# end`	Exits connect configuration mode and returns to privileged EXEC mode.
Step 14	**pseudowire-class** [*pw-class-name*] **Example:** `Device(config)# pseudowire-class l2tpv3`	Specifies the name of a Layer 2 pseudowire class and enters pseudowire class configuration mode.
Step 15	**encapsulation l2tpv3** **Example:** `Device(config-pw)# encapsulation l2tpv3`	Specifies the tunneling encapsulation as L2TPv3.
Step 16	**ip local interface loopback** *loopback id* **Example:** `Device(config-pw)# ip local interface Loopback0`	Specifies the local loopback interface.
Step 17	**connect** *connection-name interface dlci* **l2transport** **Example:** `Device(config-pw)# connect fr1 serial5/0 1000 l2transport`	Defines connections between Frame Relay Permanent Virtual Circuits (PVCs) and enters connect configuration mode. • Using the **l2transport** keyword specifies that the PVC is not a locally switched PVC, but is tunneled over the backbone network. • The *connection-name* argument is a text string that you provide. • The *interface* argument is the interface on which a PVC connection is defined. • The *dlci* argument is the DLCI number of the PVC that is connected.
Step 18	**xconnect** *peer-router-id vcid* **encapsulation l2tpv3 pw-class l2tpv3** **Example:** `Device(config-xconnect-conn-config)# xconnect 198.51.100.2 123 encapsulation l2tpv3 pw-class l2tpv3`	Creates the VC to transport the Layer 2 packets. • In a DLCI-to-DLCI connection type, Frame Relay over L2TPv3 uses the **xconnect** command in connect configuration mode.
Step 19	**end** **Example:** `Device(config-xconnect-conn-config)# end`	Exits connect configuration mode and returns to privileged EXEC mode.

Verifying Frame Relay over L2TPv3 Configuration

You can use **show** commands to view information about a Frame Relay over L2TPv3 configuration.

SUMMARY STEPS

1. **show xconnect all detail**
2. **show frame-relay pvc**
3. **show connection**

DETAILED STEPS

Step 1 **show xconnect all detail**

The following example is sample output of the **show xconnect all detail** command:

Example:

```
Device# show xconnect all detail

Legend:    XC ST=Xconnect State  S1=Segment1 State  S2=Segment2 State
  UP=Up       DN=Down            AD=Admin Down      IA=Inactive
  SB=Standby  HS=Hot Standby     RV=Recovering      NH=No Hardware

XC ST  Segment 1                         S1 Segment 2                         S2
------+---------------------------------+--+---------------------------------+--
UP pri   ac Se0/2/0:0:16(FR DLCI)        UP l2tp 22.2.2.2:100                 UP
            Interworking: L2L                    Session ID: 306532470
                                                 Tunnel ID: 1381396806
                                                 Protocol State: UP
                                                 Remote Circuit State: UP
                                                 pw-class: fr_fr
UP pri   ac Se0/2/0:0:17(FR DLCI)        UP l2tp 22.2.2.2:101                 UP
            Interworking: Eth                    Session ID: 1373339282
                                                 Tunnel ID: 1381396806
                                                 Protocol State: UP
                                                 Remote Circuit State: UP
                                                 pw-class: fr_eth
```

Step 2 **show frame-relay pvc**

The following example is sample output of the **show frame-relay pvc** command:

Example:

```
Device# show frame-relay pvc

pseudowire100001 is up, VC status is up PW type: Ethernet
  Create time: 5d20h, last status change time: 5d20h
    Last label FSM state change time: 5d20h
  Destination address: 2.1.1.2 VC ID: 1234000
    Output interface: Et0/0, imposed label stack {2001}
  Preferred path: not configured
  Default path: active
  Next hop: 20.0.0.2
Member of xconnect service Et1/0.1-1001, group right
  Associated member Et1/0.1 is up, status is up
  Interworking type is Ethernet
  Service id: 0x6d000002
Signaling protocol: LDP, peer 2.1.1.2:0 up
```

```
  Targeted Hello: 1.1.1.1(LDP Id) -> 2.1.1.2, LDP is UP
  Graceful restart: not configured and not enabled
  Non stop routing: not configured and not enabled
  PWid FEC (128), VC ID: 1234000
  Status TLV support (local/remote) : enabled/supported
    LDP route watch : enabled
    Label/status state machine : established, LruRru
    Local dataplane status received : No fault
    BFD dataplane status received : Not sent
    BFD peer monitor status received : No fault
    Status received from access circuit : No fault
    Status sent to access circuit : No fault
    Status received from pseudowire i/f : No fault
    Status sent to network peer : No fault
    Status received from network peer : No fault
    Adjacency status of remote peer : No fault
Sequencing: receive disabled, send disabled
Bindings
Parameter    Local                          Remote
------------ ------------------------------ ------------------------------
Label        2007                           2001
Group ID     0                              6
Interface
MTU          1500                           1500
Control word on (configured: autosense)     on
PW type      Ethernet                       Ethernet
VCCV CV type 0x12                           0x12
               LSPV [2], BFD/Raw [5]          LSPV [2], BFD/Raw [5]
VCCV CC type 0x07                           0x07
             CW [1], RA [2], TTL [3]        CW [1], RA [2], TTL [3]
Status TLV   enabled                        supported
Dataplane:
  SSM segment/switch IDs: 12309/4115 (used), PWID: 1
Rx Counters
  106563 input transit packets, 9803650 bytes
  0 drops, 0 seq err
Tx Counters
  0 output transit packets, 0 bytes
  0 drops
```

Step 3 **show connection**

The following example is sample output of the **show connection** command:

Example:

```
Device# show connection

ID   Name            Segment 1              Segment 2                State
================================================================================
1    fr_fr           Se0/2/0:0 16           22.2.2.2 100             UP
2    fr_eth          Se0/2/0:0 17           22.2.2.2 101             UP

--------------------------
```

Configuration Examples for Frame Relay over L2TPv3

Example: Frame Relay over L2TPv3 with LMI

The following example shows how to configure Frame Relay over L2TPv3 with Local Management Interface (LMI) enabled:

PE1 device	CE1 device

configure terminal
 interface Serial 0/2/0:0
  no ip address
  encapsulation frame-relay
!
keepalive 15
frame-relay lmi-type cisco
	configure terminal
	 interface Serial 1/0:0
	  no ip address
	  encapsulation frame-relay
	  frame-relay intf-type dce
	!
	keepalive 15
	frame-relay lmi-type cisco
	interface Serial 1/0:0.100 point-to-point
	
	 ip address 198.51.100.33 255.255.255.0
	 frame-relay interface-dlci 16

Examples: Frame Relay over L2TPv3 without LMI

The following example shows how to configure Frame Relay DLCI-to-Frame Relay DLCI over L2TPv3 without Local Management Interface (LMI) enabled:

PE1 device	CE1 device

configure terminal
 interface Serial 0/1/0
  encapsulation frame-relay
!
pseudowire-class fr_l2tpv3
 encapsulation l2tpv3
 ip local interface Loopback0
!
connect FR Serial 0/1/0 100 l2transport
 xconnect 198.51.100.2 100 encapsulation l2tpv3
pw-class fr_l2tpv3
	configure terminal
	 interface Serial 0/0/0
	  encapsulation frame-relay
	  exit
	!
	interface Serial 0/0/0.100 point-to-point
	
	 ip address 198.51.100.22 255.255.255.0
	 frame-relay interface-dlci 100

The following example shows how to configure Frame Relay DLCI-to-Ethernet Interworking over L2TPv3 without LMI enabled:

<table>
<tr><th>PE1 device</th><th>CE1 device</th></tr>
<tr><td><pre>configure terminal
 pseudowire-class fr_eth
 encapsulation l2tpv3
 interworking ethernet
 ip local interface Loopback0
!
connect FR-Eth Serial 0/1/0 500 l2transport
 xconnect 198.51.100.27 500 encapsulation l2tpv3
 pw-class fr_eth</pre></td><td><pre>configure terminal
 interface Serial 0/0/0.500 point-to-point

 frame-relay interface-dlci 500
!
interface BVI 200
 ip address 198.51.100.29 255.255.255.0</pre></td></tr>
</table>

Additional References for Frame Relay over L2TPv3

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference
Configuring Frame Relay over MPLS	*Configuring Frame Relay over MPLS*
MPLS Layer 2 VPNs Configuration Guide	*MPLS Layer 2 VPNs Configuration Guide*

Standards and RFCs

Standard/RFC	Title
RFC 2427	*Multiprotocol Interconnect over Frame Relay*
RFC 4591	*Frame Relay over Layer 2 Tunneling Protocol Version 3 (L2TPv3)*

MIBs

MIB	MIBs Link
• Cisco Frame Relay MIB (CISCO-FRAME-RELAY-MIB.my) • Interfaces MIB (IF-MIB.my)	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register on Cisco.com.	http://www.cisco.com/support

Feature Information for Frame Relay over L2TPv3

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 41: Feature Information for Frame Relay over L2TPv3

Feature Name	Releases	Feature Information
Frame Relay over L2TPv3	Cisco IOS XE Release 3.12S	This feature enables Frame Relay switching over Layer 2 Tunnel Protocol Version 3 (L2TPv3). The feature works with like interfaces and disparate interfaces (L2VPN interworking).

CHAPTER 28

Loop-Free Alternate Fast Reroute with L2VPN

The Loop-Free Alternate (LFA) Fast Reroute (FRR) with Layer 2 Virtual Private Network (L2VPN) feature minimizes packet loss due to link or node failure.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Loop-Free Alternate Fast Reroute with L2VPN

- Load balancing is not supported
- Time-division multiplexing (TDM) pseudowire is not supported
- Virtual Private LAN Services (VPLS) is not supported
- The Virtual Private Wire Services (VPWS) scale number might change

Information About Loop-Free Alternate Fast Reroute with L2VPN

L2VPN Over Loop-Free Alternate Fast Reroute

The Loop-Free Alternate (LFA) Fast Reroute (FRR) feature offers an alternative to the MPLS Traffic Engineering Fast Reroute feature to minimize packet loss due to link or node failure. It introduces LFA FRR support for L2VPNs and Virtual Private Wire Services (VPWS), providing the following benefits:

- Same level of protection from traffic loss
- Simplified configuration
- Link and node protection
- Link and path protection
- LFA (loop-free alternate) paths
- Support for both IP and Label Distribution Protocol (LDP) core

LFA FRR enables a backup route to avoid traffic loss if a network fails. The backup routes (repair paths) are precomputed and installed in the router as the backup for the primary paths. After the router detects a link or adjacent node failure, it switches to the backup path to avoid traffic loss.

How to Configure Loop-Free Alternate Fast Reroute with L2VPN

To enable loop-free alternate fast reroute support for L2VPNs and VPWS, you must configure LFA FRR for the routing protocol. No additional configuration tasks are necessary. See one of the following documents, depending on the routing protocol:

- IS-IS Remote Loop-Free Alternate Fast Reroute in the *IP Routing: ISIS Configuration Guide*
- OSPFv2 Loop-Free Alternate Fast Reroute in the *IP Routing: OSPF Configuration Guide*
- OSPF IPv4 Remote Loop-Free Alternate IP Fast Reroute in the *IP Routing: OSPF Configuration Guide*

Verifying Loop-Free Alternate Fast Reroute with L2VPN

Use one or more of the following commands to verify the LFA FRR configuration:

SUMMARY STEPS

1. **show ip cef** *network-prefix* **internal**
2. **show mpls infrastructure lfd pseudowire internal**
3. **show platform hardware pp active feature cef database ipv4** *network-prefix*

DETAILED STEPS

Step 1 **show ip cef** *network-prefix* **internal**

Example:

```
show ip cef 16.16.16.16 internal
```

Displays entries in the Cisco Express Forwarding (CEF) Forwarding Information Base (FIB).

Step 2 **show mpls infrastructure lfd pseudowire internal**

Example:

```
show mpls infrastructure lfd pseudowire internal
```

Displays information about the Label Forwarding Database (LFD) and pseudowires.

Step 3 **show platform hardware pp active feature cef database ipv4** *network-prefix*

Example:

```
show platform hardware pp active feature cef database ipv4 16.16.16.16/32
```

Displays information about the CEF database.

Configuration Examples for Loop-Free Alternate Fast Reroute with L2VPN

Example: Verifying LFA FRR with L2VPN

show ip cef internal

The following example shows the configuration of LFA FRR for OSPF:

```
router ospf 1
 router-id 17.17.17.17
 fast-reroute per-prefix enable prefix-priority low
 network 3.3.3.0 0.0.0.255 area 1
 network 6.6.6.0 0.0.0.255 area 1
 network 7.7.7.0 0.0.0.255 area 1
 network 17.17.17.17 0.0.0.0 area 1
```

show ip cef internal

The following is sample output from the **show ip cef internal** command:

```
Device# show ip cef 16.16.16.16 internal
16.16.16.16/32, epoch 2, RIB[I], refcount 7, per-destination sharing
  sources: RIB, RR, LTE
  feature space:
   IPRM: 0x00028000
   Broker: linked, distributed at 1st priority
```

```
    LFD: 16.16.16.16/32 1 local label
    local label info: global/17
         contains path extension list
         disposition chain 0x3A3C1DF0
         label switch chain 0x3A3C1DF0
   subblocks:
    1 RR source [no flags]
     non-eos chain [16|44]
   ifnums:
    GigabitEthernet0/0/2(9): 7.7.7.2
    GigabitEthernet0/0/7(14): 7.7.17.9
   path 35D61070, path list 3A388FA8, share 1/1, type attached nexthop, for IPv4, flags
has-repair
     MPLS short path extensions: MOI flags = 0x20 label 16
   nexthop 7.7.7.2 GigabitEthernet0/0/2 label [16|44], adjacency IP adj out of
GigabitEthernet0/0/2, addr 7.7.7.2 35E88520
     repair: attached-nexthop 7.7.17.9 GigabitEthernet0/0/7 (35D610E0)
   path 35D610E0, path list 3A388FA8, share 1/1, type attached nexthop, for IPv4, flags
repair, repair-only
   nexthop 7.7.17.9 GigabitEthernet0/0/7, repair, adjacency IP adj out of GigabitEthernet0/0/7,
  addr 7.7.17.9 3A48A4E0
   output chain: label [16|44]
   FRR Primary (0x35D10F60)
   <primary:  TAG adj out of GigabitEthernet0/0/2, addr 7.7.7.2 35E88380>
   <repair:  TAG adj out of GigabitEthernet0/0/7, addr 7.7.17.9 3A48A340>
Rudy17#show mpls infrastructure lfd pseudowire internal
PW ID: 1VC ID: 4, Nexthop address: 16.16.16.16
SSM Class: SSS HW
Segment Count: 1
VCCV Types Supported:  cw ra ttl
Imposition details:
 Label stack {22 16}, Output interface: Gi0/0/2
 Preferred path: not configured
 Control Word: enabled, Sequencing: disabled
 FIB Non IP entry: 0x35D6CEEC
 Output chain:  AToM Imp (locks 4) label 22 label [16|44]
  FRR Primary (0x35D10F60)
  <primary:  TAG adj out of GigabitEthernet0/0/2, addr 7.7.7.2 35E88380>
Disposition details:
 Local label: 16
 Control Word: enabled, Sequencing: disabled
 SSS Switch: 3976200193
 Output chain:  mpls_eos( connid router-alert AToM Disp (locks 5)/ drop)
```

show mpls infrastructure lfd pseudowire internal

The following is sample output from the **show mpls infrastructure lfd pseudowire internal** command:

```
Device# show mpls infrastructure lfd pseudowire internal
PW ID: 1VC ID: 4, Nexthop address: 16.16.16.16
SSM Class: SSS HW
Segment Count: 1
VCCV Types Supported:  cw ra ttl
Imposition details:
 Label stack {22 16}, Output interface: Gi0/0/2
 Preferred path: not configured
 Control Word: enabled, Sequencing: disabled
 FIB Non IP entry: 0x35D6CEEC
 Output chain:  AToM Imp (locks 4) label 22 label [16|44]
  FRR Primary (0x35D10F60)
  <primary:  TAG adj out of GigabitEthernet0/0/2, addr 7.7.7.2 35E88380>
```

```
Disposition details:
 Local label: 16
 Control Word: enabled, Sequencing: disabled
 SSS Switch: 3976200193
 Output chain:  mpls_eos( connid router-alert AToM Disp (locks 5)/ drop)
```

show platform hardware pp active feature cef database

The following is sample output from the **show platform hardware pp active feature cef database** command:

```
Device# show platform hardware pp active feature cef database ipv4 16.16.16.16/32
=== CEF Prefix ===
16.16.16.16/32 -- next hop: UEA Label OCE (PI:0x104abee0, PD:0x10e6b9c8)
                Route Flags: (0)
                Handles (PI:0x104ab6e0) (PD:0x10e68140)

  HW Info:
    TCAM handle: 0x0000023f    TCAM index: 0x0000000d
    FID index  : 0x0000f804    EAID      : 0x0000808a
    MET        : 0x0000400c    FID Count : 0x00000000

=== Label OCE ===
  Label flags: 4
  Num Labels: 1
  Num Bk Labels: 1
  Out Labels: 16
  Out Backup Labels: 44
  Next OCE Type: Fast ReRoute OCE; Next OCE handle: 0x10e6f428

=== FRR OCE ===
  FRR type         : IP FRR
  FRR state        : Primary
  Primary IF's gid : 3
  Primary FID      : 0x0000f801
  FIFC entries     : 32
  PPO handle       : 0x00000000
  Next OCE         : Adjacency (0x10e63b38)
  Bkup OCE         : Adjacency (0x10e6e590)

=== Adjacency OCE ===
  Adj State: COMPLETE(0)   Address: 7.7.7.2
  Interface: GigabitEthernet0/0/2   Protocol: TAG
  mtu:1500, flags:0x0, fixups:0x0, encap_len:14
  Handles (adj_id:0x00000039) (PI:0x1041d410) (PD:0x10e63b38)
  Rewrite Str: d0:c2:82:17:8a:82:d0:c2:82:17:f2:02:88:47

  HW Info:
    FID index: 0x0000f486    EL3 index: 0x00001003    EL2 index: 0x00000000
    El2RW    : 0x00000107    MET index: 0x0000400c    EAID     : 0x00008060
    HW ADJ FLAGS: 0x40
    Hardware MAC Rewrite Str: d0:c2:82:17:8a:82:08:00:40:00:0d:02

=== Adjacency OCE ===
  Adj State: COMPLETE(0)   Address: 7.7.17.9
  Interface: GigabitEthernet0/0/7   Protocol: TAG
  mtu:1500, flags:0x0, fixups:0x0, encap_len:14
  Handles (adj_id:0x00000012) (PI:0x104acbd0) (PD:0x10e6e590)
  Rewrite Str: d0:c2:82:17:c9:83:d0:c2:82:17:f2:07:88:47

  HW Info:
    FID index: 0x0000f49d    EL3 index: 0x00001008    EL2 index: 0x00000000
```

```
El2RW    : 0x00000111    MET index: 0x00004017    EAID     : 0x0000807d
HW ADJ FLAGS: 0x40
Hardware MAC Rewrite Str: d0:c2:82:17:c9:83:08:00:40:00:0d:07
```

Example: Configuring Remote LFA FRR with VPLS

Example: Configuration of Remote LFA FRR with Interior Gateway Protocol (IGP)

```
router isis hp
 net 49.0101.0000.0000.0802.00
 is-type level-2-only
 ispf level-2
 metric-style wide
 fast-flood
 set-overload-bit on-startup 180
 max-lsp-lifetime 65535
 lsp-refresh-interval 65000
 spf-interval 5 50 200
 prc-interval 5 50 200
 lsp-gen-interval 5 5 200
 no hello padding
 log-adjacency-changes
 nsf cisco
 fast-reroute per-prefix level-1 all
 fast-reroute per-prefix level-2 all
 fast-reroute remote-lfa level-1 mpls-ldp
 fast-reroute remote-lfa level-2 mpls-ldp
 passive-interface Loopback0
 mpls ldp sync
 mpls traffic-eng router-id Loopback0
 mpls traffic-eng level-2
```

Example: Configuration of Remote LFA FRR with VPLS at the interface level.

```
!
interface GigabitEthernet0/3/3
 ip address 198.51.100.1 255.255.255.0
 ip router isis hp
 logging event link-status
 load-interval 30
 negotiation auto
 mpls ip
 mpls traffic-eng tunnels
 isis network point-to-point
end
!
```

Example: Configuration of remote LFA FRR with VPLS at the global level.

```
!
l2 vfi Test-2000 manual
 vpn id 2010
 bridge-domain 2010
 neighbor 192.0.2.1 encapsulation mpls
!
```

Example: Configuration of remote LFA FRR with VPLS at Access side.

```
!
interface TenGigabitEthernet0/2/0
 no ip address
 service instance trunk 1 ethernet
  encapsulation dot1q 12-2012
  rewrite ingress tag pop 1 symmetric
  bridge-domain from-encapsulation
 !
```

Example: Verifying Remote LFA FRR with VPLS

show ip cef internal

The following is sample output from the **show ip cef internal** command:

```
Router# show ip cef 198.51.100.2/32 internal

198.51.100.2/32, epoch 2, RIB[I], refcount 7, per-destination sharing
  sources: RIB, RR, LTE
  feature space:
   IPRM: 0x00028000
   Broker: linked, distributed at 1st priority
   LFD: 198.51.100.2/32 1 local label
   local label info: global/2033
        contains path extension list
        disposition chain 0x46764E68
        label switch chain 0x46764E68
  subblocks:
   1 RR source [heavily shared]
    non-eos chain [explicit-null|70]
  ifnums:
   TenGigabitEthernet0/1/0(15): 192.0.2.10
   MPLS-Remote-Lfa2(46)
  path 44CE1290, path list 433CF8C0, share 1/1, type attached nexthop, for IPv4, flags
has-repair
    MPLS short path extensions: MOI flags = 0x21 label explicit-null
  nexthop 192.0.2.10 TenGigabitEthernet0/1/0 label [explicit-null|70], adjacency IP adj out
 of TenGigabitEthernet0/1/0, addr 192.0.2.10 404B3960
    repair: attached-nexthop 192.0.2.1 MPLS-Remote-Lfa2 (44CE1300)
  path 44CE1300, path list 433CF8C0, share 1/1, type attached nexthop, for IPv4, flags
repair, repair-only
  nexthop 192.0.2.1 MPLS-Remote-Lfa2, repair, adjacency IP midchain out of MPLS-Remote-Lfa2
 404B3B00
  output chain: label [explicit-null|70]
  FRR Primary (0x3E25CA00)
  <primary:  TAG adj out of TenGigabitEthernet0/1/0, addr 192.168.101.22 404B3CA0>
  <repair:  TAG midchain out of MPLS-Remote-Lfa2 404B37C0 label 37 TAG adj out of
GigabitEthernet0/3/3, addr 192.0.2.14 461B2F20>
```

show ip cef detail

The following is sample output from the **show ip cef detail** command:

```
Router# show ip cef 198.51.100.2/32 detail

198.51.100.2/32, epoch 2
```

```
  local label info: global/2033
  1 RR source [heavily shared]
  nexthop 192.0.2.14 TenGigabitEthernet0/1/0 label [explicit-null|70]
    repair: attached-nexthop 192.0.2.1 MPLS-Remote-Lfa2
  nexthop 192.0.2.1 MPLS-Remote-Lfa2, repair
!
```

show platform hardware pp active feature cef databas

The following is sample output from the **show platform hardware pp active feature cef database** command:

```
Router# show platform hardware pp active feature cef database ipv4 198.51.100.2/32

=== CEF Prefix ===
198.51.100.2/32 -- next hop: UEA Label OCE (PI:0x10936770, PD:0x12dd1cd8)
                Route Flags: (0)
                Handles (PI:0x109099c8) (PD:0x12945968)

  HW Info:
    TCAM handle: 0x00000266    TCAM index: 0x00000015
    FID index  : 0x00008e7f    EAID      : 0x0001d7c4
    MET        : 0x0000401c    FID Count : 0x00000000
=== Label OCE ===
  Label flags: 4
  Num Labels: 1
  Num Bk Labels: 1
  Out Labels: 0
  Out Backup Labels: 70
=== FRR OCE ===
  FRR type         : IP FRR
  FRR state        : Primary
  Primary IF's gid : 52
  Primary FID      : 0x00008cb6
  FIFC entries     : 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0
  PPO handle       : 0x00000000
  Next OCE         : Adjacency (0x130e0df0)
  Bkup OCE         : Adjacency (0x130de608)

=== Adjacency OCE ===
  Adj State: COMPLETE(0)   Address: 192.168.101.22
  Interface: TenGigabitEthernet0/1/0   Protocol: TAG
  mtu:1500, flags:0x0, fixups:0x0, encap_len:14
  Handles (adj_id:0x000016ac) (PI:0x1090cc10) (PD:0x130e0df0)
  Rewrite Str: 18:33:9d:3d:83:10:c8:f9:f9:8d:04:10:88:47
HW Info:
    FID index: 0x00008e7e    EL3 index: 0x00001034    EL2 index: 0x00000000
    El2RW    : 0x0000010d    MET index: 0x00004012    EAID     : 0x0001d7c1
    HW ADJ FLAGS: 0x40
    Hardware MAC Rewrite Str: 18:33:9d:3d:83:10:08:00:40:00:0d:10
=== Adjacency OCE ===
  Adj State: COMPLETE(0)   Address: 0
  Interface: MPLS-Remote-Lfa2   Protocol: TAG
  mtu:17940, flags:0x40, fixups:0x0, encap_len:0
  Handles (adj_id:0xf80002e8) (PI:0x10da2150) (PD:0x130de608)
  Rewrite Str:

  HW Info:
    FID index: 0x00008ca8    EL3 index: 0x0000101c    EL2 index: 0x00000000
    El2RW    : 0x00000003    MET index: 0x00004024    EAID     : 0x0001d7cb
    HW ADJ FLAGS: 0x40
```

```
    Hardware MAC Rewrite Str: 00:00:00:00:00:00:00:00:00:00:00:00

=== Label OCE ===
  Label flags: 4
  Num Labels: 1
Num Bk Labels: 1
  Out Labels: 37
  Out Backup Labels: 37
  Next OCE Type: Adjacency; Next OCE handle: 0x12943a00
=== Adjacency OCE ===
  Adj State: COMPLETE(0)   Address: 30.1.1.1
  Interface: GigabitEthernet0/3/3   Protocol: TAG
  mtu:1500, flags:0x0, fixups:0x0, encap_len:14
  Handles (adj_id:0x0000378e) (PI:0x10909738) (PD:0x12943a00)
  Rewrite Str: c8:f9:f9:8d:01:b3:c8:f9:f9:8d:04:33:88:47

  HW Info:
    FID index: 0x00008c78    EL3 index: 0x0000101c    EL2 index: 0x00000000
    El2RW    : 0x00000109    MET index: 0x0000400e    EAID     : 0x0001cf4b
    HW ADJ FLAGS: 0x40
    Hardware MAC Rewrite Str: c8:f9:f9:8d:01:b3:08:00:40:00:0d:33
```

show mpls l2transport detail

The following is sample output from the **show mpls l2transport detail** command:

```
Router# show mpls l2transport vc 2000 detail

Local interface: VFI Test-1990 vfi up
  Interworking type is Ethernet
  Destination address: 192.0.2.1, VC ID: 2000, VC status: up
    Output interface: Te0/1/0, imposed label stack {0 2217}
    Preferred path: not configured
    Default path: active
    Next hop: 192.51.100.22
  Create time: 1d08h, last status change time: 1d08h
    Last label FSM state change time: 1d08h
  Signaling protocol: LDP, peer 192.0.51.1:0 up
    Targeted Hello: 192.51.100.2(LDP Id) -> 192.51.100.200, LDP is UP
    Graceful restart: configured and enabled
    Non stop routing: not configured and not enabled
    Status TLV support (local/remote)    : enabled/supported
      LDP route watch                    : enabled
      Label/status state machine         : established, LruRru
      Last local dataplane   status rcvd: No fault
      Last BFD dataplane     status rcvd: Not sent
      Last BFD peer monitor  status rcvd: No fault
      Last local AC  circuit status rcvd: No fault
      Last local AC  circuit status sent: No fault
      Last local PW i/f circ status rcvd: No fault
      Last local LDP TLV     status sent: No fault
      Last remote LDP TLV    status rcvd: No fault
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Loop-Free Alternate Fast Reroute with L2VPN

Table 42: Feature Information for Loop-Free Alternate Fast Reroute with L2VPN

Feature Name	Releases	Feature Information
Loop-Free Alternate Fast Reroute with L2VPN	15.3(2)S Cisco IOS XE Release 3.9S Cisco IOS XE Release 3.10 S	This feature introduces loop-free alternate (LFA) fast reroute (FRR) support for Layer 2 VPN (L2VPN) and Virtual Private Wire Services (VPWS) to minimize packet loss due to link or node failure. No commands were introduced or modified. In Cisco IOS XE Release 3.9S, support was added for the Cisco ASR 903 Router. In Cisco IOS XE Release 3.10S, Remote LFA FRR is supported on ATM (IMA) and TDM pseudowires for the Cisco ASR 903 Router. In Cisco IOS XE Release 3.10S, Remote LFA FRR is supported over VPLS for Cisco ASR 903 Router.