Implementing Point to Point Layer 2 services

This module provides the conceptual and configuration information for Point to Point Layer 2 services on Cisco IOS XR software.

Note The Point to Point Layer 2 services are also called as MPLS Layer 2 VPNs.

Note To locate documentation for other commands that might appear while executing a configuration task, search online in the Cisco IOS XR software master command index.

Feature History for Implementing Point to Point Layer 2 services Configuration Module

Release	Modification
Release 3.4.0	This feature was introduced.
Release 3.4.1	Support was added for: • Virtual Circuit Connection Verification (VCCV) on L2VPN • Layer 2 VPN (L2VPN) Quality of Service (QoS) for Ethernet-over-MPLS (EoMPLS)
Release 3.5.0	Support was added for: • EoMPLS Inter-AS mode • Mac-in-Mac protocol
Release 3.6.0	Support was added for: • Ethernet Remote Port Shutdown • Preferred Tunnel Path

Release	Modification
Release 3.7.0	Support was added for ATM over MPLS (ATMoMPLS) with Layer 2VPN capability.
Release 3.8.0	Support was added for QinQ mode and QinAny mode for EoMPLS.
Release 3.9.0	Support was added for the Generic Routing Encapsulation (GRE) feature.
Release 4.2.0	Support was added for the Load Balancing with Flow Aware Transport pseudowires (FAT PW) feature.

Prerequisites for Implementing Point to Point Layer 2 Services

To perform these configuration tasks, your Cisco IOS XR software system administrator must assign you to a user group associated with a task group that includes the corresponding command task IDs. All command task IDs are listed in individual command references and in the *Cisco IOS XR Task ID Reference Guide*.

If you need assistance with your task group assignment, contact your system administrator.

Information About Implementing Point to Point Layer 2 Services

To implement Point to Point Layer 2 Services, you should understand these concepts:

L2VPN Overview

Layer 2 VPN (L2VPN) emulates the behavior of a LAN across an IP or MPLS-enabled IP network allowing Ethernet devices to communicate with each other as they would when connected to a common LAN segment.

As Internet service providers (ISPs) look to replace their Asynchronous Transfer Mode (ATM) infrastructures with an IP infrastructure, there is a need for to provide standard methods of using an IP infrastructure to provide a serviceable L2 interface to customers; specifically, to provide standard ways of using an IP infrastructure to provide virtual circuits between pairs of customer sites.

Building a L2VPN system requires coordination between the ISP and the customer. The ISP provides L2 connectivity; the customer builds a network using data link resources obtained from the ISP. In an L2VPN service, the ISP does not require information about a the customer's network topology, policies, routing information, point-to-point links, or network point-to-point links from other ISPs.

The ISP requires provider edge (PE) routers with the following capabilities:

- Encapsulation of L2 protocol data units (PDU) into Layer 3 (L3) packets.
- Interconnection of any-to-any L2 transports.
- Emulation of L2 quality-of-service (QoS) over a packet switch network.
- Ease of configuration of the L2 service.
- Support for different types of tunneling mechanisms (MPLS, L2TPv3, IPSec, GRE, and others).
- L2VPN process databases include all information related to circuits and their connections.

ATMoMPLS with L2VPN Capability

These topics describe the ATM over MPLS (ATMoMPLS) with L2VPN feature:

ATMoMPLS with L2VPN Overview

The ATMoMPLS feature supports ATM Adaptation Layer 5 (AAL5) transport. ATMoMPLS is a type of Layer 2 point-to-point connection over an MPLS core. ATMoMPLS and ATM local switching are supported only for ATM-to-ATM interface-to-interface switching combinations.

To implement the ATMoMPLS feature, the Cisco CRS Router plays the role of provider edge (PE) router at the edge of a provider network in which customer edge (CE) devices are connected to the Cisco CRS Router.

Layer 2 Local Switching Overview

Local switching lets you to switch Layer 2 data between two interfaces of the same type (for example, ATM-to-ATM, or Frame Relay-to-Frame Relay) or between interfaces of different types (for example, Frame Relay to ATM) on the same router, over an IP core network. The interfaces are on the same line card or on two different cards. During these types of switching, Layer 2 address is used instead of the Layer 3 address.

In addition, same-port local switching lets you to switch Layer 2 data between two circuits on the same interface.

ATM Adaptation Layer 5

AAL5 lets you transport AAL5 PDUs from various customers over an MPLS backbone. ATM AAL5 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 AAL5 PDUs by configuring the provider edge (PE) routers at both ends of the MPLS backbone.

To transport AAL5 PDUs over MPLS, a virtual circuit is set up from the ingress PE router to the egress PE router. This virtual circuit transports the AAL5 PDUs from one PE router to the other. Each AAL5 PDU is transported as a single packet.

Virtual Circuit Connection Verification on L2VPN

Virtual Circuit Connection Verification (VCCV) is an L2VPN Operations, Administration, and Maintenance (OAM) feature that allows network operators to run IP-based provider edge-to-provider edge (PE-to-PE) keepalive protocol across a specified pseudowire to ensure that the pseudowire data path forwarding does not contain any faults. The disposition PE receives VCCV packets on a control channel, which is associated with the specified pseudowire. The control channel type and connectivity verification type, which are used for VCCV, are negotiated when the pseudowire is established between the PEs for each direction.

Two types of packets can arrive at the disposition egress:

- Type 1—Specifies normal Ethernet-over-MPLS (EoMPLS) data packets.
- Type 2—Specifies VCCV packets.

Cisco CRS Router supports Label Switched Path (LSP) VCCV Type 1, which uses an inband control word if enabled during signaling. The VCCV echo reply is sent as IPv4 that is the reply mode in IPv4. The reply is forwarded as IP, MPLS, or a combination of both.

VCCV pings counters that are counted in MPLS forwarding on the egress side. However, on the ingress side, they are sourced by the route processor and do not count as MPLS forwarding counters.

Ethernet over MPLS

Ethernet-over-MPLS (EoMPLS) provides a tunneling mechanism for Ethernet traffic through an MPLS-enabled L3 core and encapsulates Ethernet protocol data units (PDUs) inside MPLS packets (using label stacking) to forward them across the MPLS network.

EoMPLS features are described in these subsections:

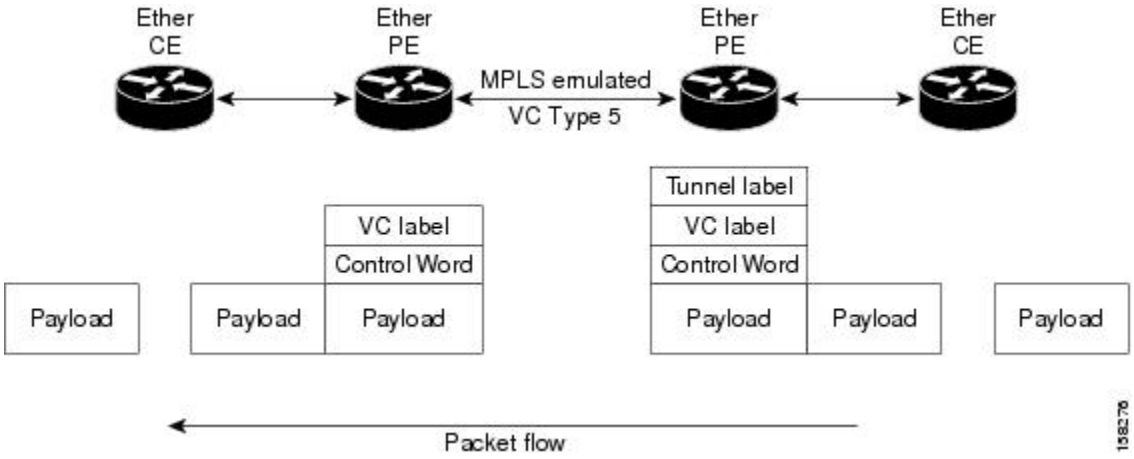
Ethernet Port Mode

In Ethernet port mode, both ends of a pseudowire are connected to Ethernet ports. In this mode, the port is tunneled over the pseudowire or, using local switching (also known as an *attachment circuit-to-attachment circuit cross-connect*) switches packets or frames from one attachment circuit (AC) to another AC attached to the same PE node.

Note L2VPN forwarding using GRE tunnels is supported in the Ethernet port mode.

The following figure provides an example of Ethernet port mode.

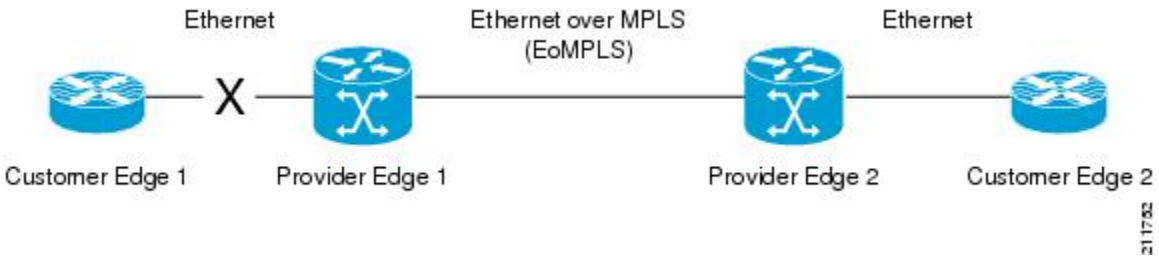
Figure 1: Ethernet Port Mode Packet Flow



Ethernet Remote Port Shutdown

Ethernet remote port shutdown provides a mechanism for the detection and propagation of remote link failure for port mode EoMPLS on a Cisco CRS Router line card. This lets a service provider edge router on the local end of an Ethernet-over-MPLS (EoMPLS) pseudowire detect a cross-connect or remote link failure and cause the shutdown of the Ethernet port on the local customer edge router. Shutting down the Ethernet port on the local customer edge router prevents or mitigates a condition where that router would otherwise lose data by forwarding traffic continuously to the remote failed link, especially if the link were configured as a static IP route (see following figure)..

Figure 2: Remote Link Outage in EoMPLS Wide Area Network



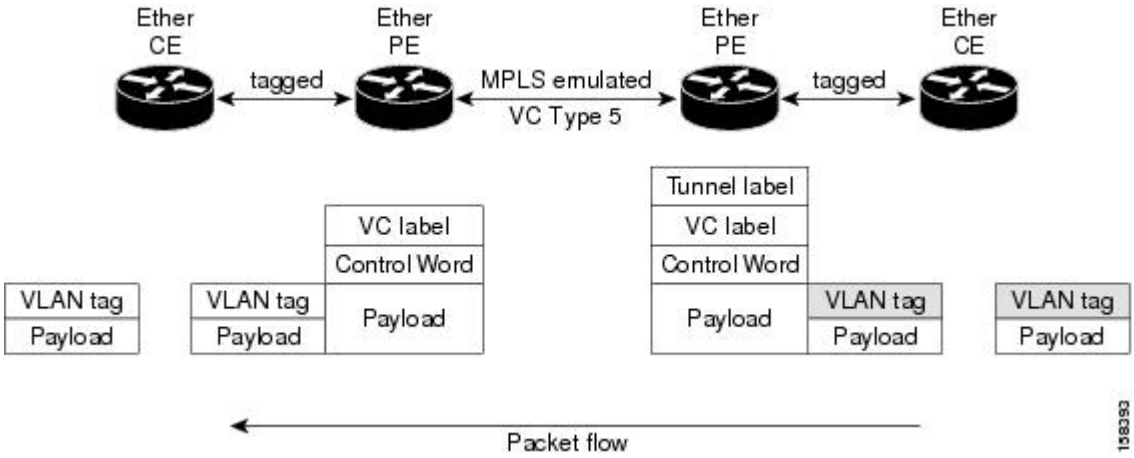
To enable this functionality, see the **l2transport propagate** command in *Cisco IOS XR MPLS Command Reference for the Cisco CRS Router*.

VLAN Mode

In VLAN mode, each VLAN on a customer-end to provider-end link can be configured as a separate L2VPN connection using virtual connection (VC) type 4 or VC type 5. VC type 4 is the default mode.

As illustrated in the following figure, the Ethernet PE associates an internal VLAN-tag to the Ethernet port for switching the traffic internally from the ingress port to the pseudowire; however, before moving traffic into the pseudowire, it removes the internal VLAN tag.

Figure 3: VLAN Mode Packet Flow



At the egress VLAN PE, the PE associates a VLAN tag to the frames coming off of the pseudowire and after switching the traffic internally, it sends out the traffic on an Ethernet trunk port.

Note Because the port is in trunk mode, the VLAN PE doesn't remove the VLAN tag and forwards the frames through the port with the added tag.

Note L2VPN forwarding using GRE tunnels is supported in the VLAN mode.

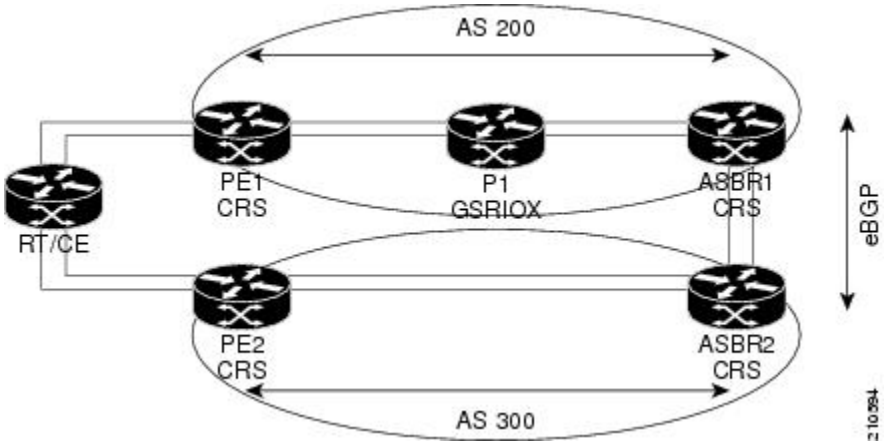
Inter-AS Mode

Inter-AS is a peer-to-peer type model that allows extension of VPNs through multiple provider or multi-domain networks. This lets service providers peer up with one another to offer end-to-end VPN connectivity over extended geographical locations.

EoMPLS support can assume a single AS topology where the pseudowire connecting the PE routers at the two ends of the point-to-point EoMPLS cross-connects resides in the same autonomous system; or multiple AS topologies in which PE routers can reside on two different ASs using iBGP and eBGP peering.

The following figure illustrates MPLS over Inter-AS with a basic double AS topology with iBGP/LDP in each AS.

Figure 4: EoMPLS over Inter-AS: Basic Double AS Topology



QinQ Mode

QinQ is an extension of 802.1Q for specifying multiple 802.1Q tags (IEEE 802.1QinQ VLAN Tag stacking). Layer 3 VPN service termination and L2VPN service transport are enabled over QinQ sub-interfaces.

The Cisco CRS Routers implement the Layer 2 tunneling or Layer 3 forwarding depending on the subinterface configuration at provider edge routers. This function only supports up to two QinQ tags on the SPA and fixed PLIM:

- Layer 2 QinQ VLANs in L2VPN attachment circuit: QinQ L2VPN attachment circuits are configured under the Layer 2 transport subinterfaces for point-to-point EoMPLS based cross-connects using both virtual circuit type 4 and type 5 pseudowires and point-to-point local-switching-based cross-connects including full interworking support of QinQ with 802.1q VLANs and port mode.
- Layer 3 QinQ VLANs: Used as a Layer 3 termination point, both VLANs are removed at the ingress provider edge and added back at the remote provider edge as the frame is forwarded.

Layer 3 services over QinQ include:

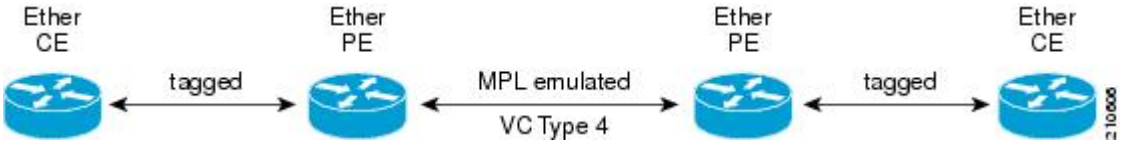
- IPv4 unicast and multicast
- IPv6 unicast and multicast
- MPLS
- Connectionless Network Service (CLNS) for use by Intermediate System-to-Intermediate System (IS-IS) Protocol

Note The Cisco CRS-1 router does not support: bundle attachment circuits and Hot Standby Router Protocol (HSRP) or Virtual Router Redundancy Protocol (VRRP) on QinQ subinterfaces.

In QinQ mode, each CE VLAN is carried into an SP VLAN. QinQ mode should use VC type 5, but VC type 4 is also supported. On each Ethernet PE, you must configure both the inner (CE VLAN) and outer (SP VLAN).

The following figure illustrates QinQ using VC type 4.

Figure 5: EoMPLS over QinQ Mode



QinAny Mode

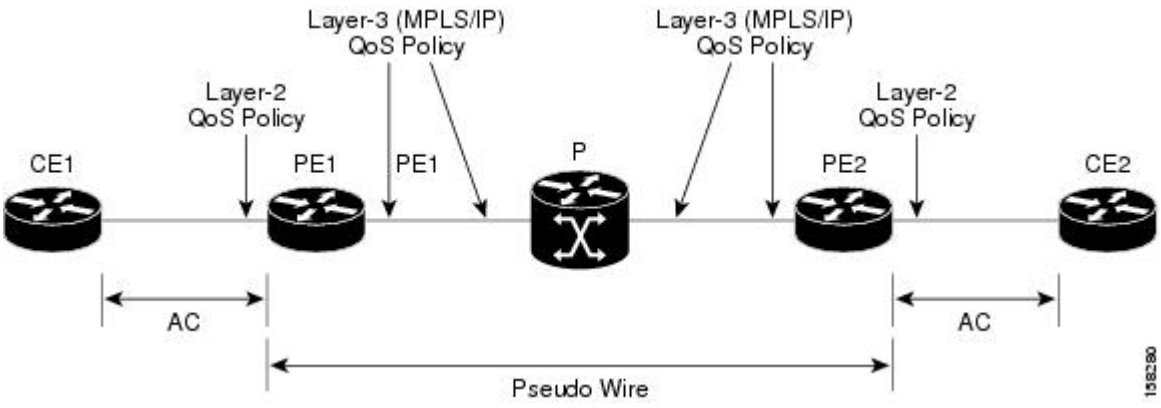
In the QinAny mode, the service provider VLAN tag is configured on both the ingress and the egress nodes of the provider edge VLAN. QinAny mode is similar to QinQ mode using a Type 5 VC, except that the customer edge VLAN tag is carried in the packet over the pseudowire, as the customer edge VLAN tag is unknown.

Quality of Service

Using L2VPN technology, you can assign a quality of service (QoS) level to both Port and VLAN modes of operation.

L2VPN technology requires that QoS functionality on PE routers be strictly L2-payload-based on the edge-facing interfaces (also know as *attachment circuits*). The following figure illustrates L2 and L3 QoS service policies in a typical L2VPN network.

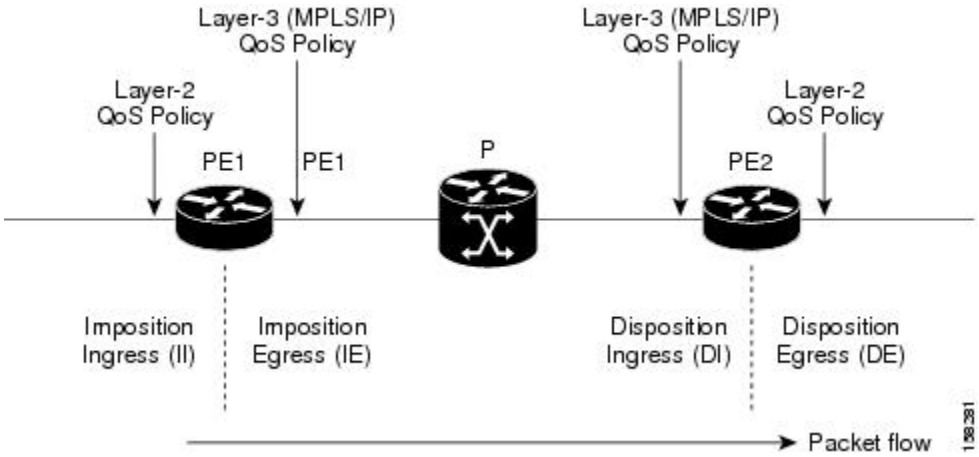
Figure 6: L2VPN QoS Feature Application



The following figure shows four packet processing paths within a provider edge device where a QoS service policy can be attached. In an L2VPN network, packets are received and transmitted on the edge-facing

interfaces as L2 packets and transported on the core-facing interfaces as MPLS (EoMPLS) or IP (L2TP) packets.

Figure 7: L2VPN QoS Reference Model



High Availability

L2VPN uses control planes in both route processors and line cards, as well as forwarding plane elements in the line cards.

Note The l2tp_mgr process does not support high availability.

The availability of L2VPN meets these requirements:

- A control plane failure in either the route processor or the line card will not affect the circuit forwarding path.
- The router processor control plane supports failover without affecting the line card control and forwarding planes.
- L2VPN integrates with existing Label Distribution Protocol (LDP) graceful restart mechanism.

Preferred Tunnel Path

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

Note

- Currently, preferred tunnel path configuration applies only to MPLS encapsulation.
- The fallback enable option is supported.

Generic Routing Encapsulation Support for L2VPN

Generic Routing Encapsulation (GRE) is a tunneling protocol that can encapsulate many types of packets to enable data transmission using a tunnel.

Ethernet over MPLS Forwarding Using GRE Tunnels

This section describes the working of the Ethernet over MPLS (EoMPLS) over GRE tunnels. The following description assumes that the GRE tunnels connect two PE routers, and Layer 2 circuits exist between the PE and CE routers.

Ingress of Encapsulation Router

To enable L2VPN forwarding over GRE tunnels, the targeted Label Distribution Protocol (LDP) neighbor session (established over the GRE tunnel interface) enables exchanging virtual circuit labels between the PE routers. LDP also installs an implicit null label to be used across the GRE tunnels. The implicit null label is a label with special semantics that an LDP can bind to an address prefix. The Layer 2 forwarding tracks the GRE tunnel or addresses, depending on the GRE tunnel.

Egress of Encapsulation Router

On the egress side of the encapsulation PE router, the following events occur:

- The Layer 2 packet passes through the regular L2VPN processing based on the local label imposed by the ingress side.
- The Layer 2 packet is encapsulated with the GRE header and outer IP header.
- The packet is transmitted based on the outer IP header information.

Ingress of Decapsulation Router

When the decapsulation router (remote PE) receives the GRE encapsulated Layer 2 packet, the following events occur:

- The packet is processed based on the outer IP header information.
- The packet is decapsulated to retrieve the inner LDP packet.

Further processing of the packet is based on the Layer 2 payload. The packet is handed over to the egress line card (that hosts the Layer 2 circuit) based on the VC label forwarding information programmed by the Layer 2 FIB.

Egress of Decapsulation Router

On the egress side of the decapsulation router, the following events occur:

- The forwarding occurs based on the Layer 2 VC label on the packet.
- A VC label lookup identifies the correct Layer 2 circuit to be used, to forward the packet towards the CE router that hosts the Layer 2 destination.

Limitations

- Fragmentation and reassembly of GRE packets are not supported.
- At GRE tunnel head end, decision to fragment packet is taken on the basis of Do-Not-Fragment (DF) flag in IPv4 header of the incoming packet. The GRE tunnel DF flag configuration is ignored while fragmenting the packet.
- The configured tunnel GRE DF flag value is inserted into transport header. However, fragmentation of GRE packet is not supported anywhere in the GRE tunnel path.
- The following features are not supported:
  - GRE tunnel in VRF domains
  - Multicast
  - V6 over GRE
  - MPLS/L3VPN over GRE
  - VPNv4 forwarding over GRE tunnels
  - 6PE/6VPE over GRE

Pseudowire Redundancy

Pseudowire redundancy allows you to configure your network to detect a failure in the network and reroute the Layer 2 service to another endpoint that can continue to provide service. This feature provides the ability to recover from a failure of either the remote provider edge (PE) router or the link between the PE and customer edge (CE) routers.

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 takes over. However, there are some parts of the network in which this rerouting mechanism does not protect against interruptions in service.

Pseudowire redundancy enables you to set up backup pseudowires. You can configure the network with redundant pseudowires and redundant network elements.

Prior to the failure of the primary pseudowire, the ability to switch traffic to the backup pseudowire is used to handle a planned pseudowire outage, such as router maintenance.

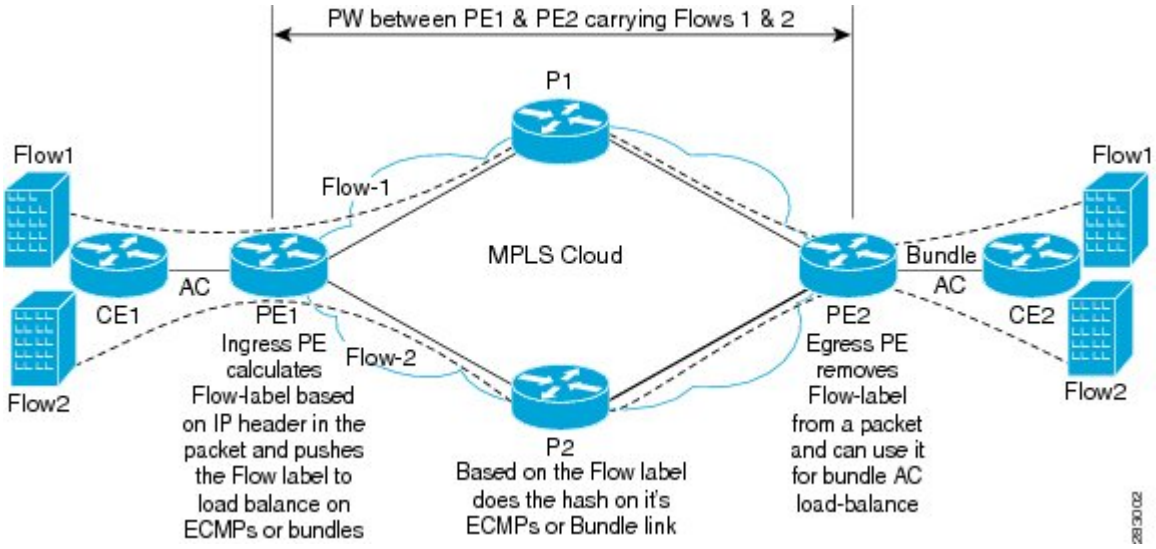
Note Pseudowire redundancy is provided only for point-to-point Virtual Private Wire Service (VPWS) pseudowires.

Flow Aware Transport Pseudowire (FAT PW)

Flow Aware Transport Pseudowires (FAT PW) are used to load-balance traffic in the core when equal cost multipaths (ECMP) are used. The flow, in this context, refers to a sequence of packets that have the same source and destination pair. The packets are transported from a source provider edge (PE) to a destination PE.

The following figure shows a FAT PW with two flows distributing over ECMPs and bundle links.

Figure 8: FAT PW with two flows distributing over ECMPs and Bundle-Links



The MPLS labels add an additional label to the stack, called the flow label, which contains the flow information of a virtual circuit (VC). A flow label is a unique identifier that distinguishes a flow within the PW, and is derived from the IP payload of a packet. The flow label contains the end of label stack (EOS) bit set and inserted after the VC label and before the control word (if any). The ingress PE calculates and forwards the flow label. The FAT PW configuration enables the flow label. The egress PE discards the flow label such that no decisions are taken based on that label.

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

L2 Traffic Tunneling with IP Load Balance Hashing for MPLS Encapsulated Packets

If the frame is IP-based, the load-balancing flow “src-dst-ip” configuration causes the Layer 2 interfaces to use the IP header for flow balancing hash calculation. If the frame is not IP-based, the MAC header is used for the hash calculation. In previous releases, for an MPLS header between the MAC and IP headers, the code would use the MAC header for the flow balancing hash.

From Release 6.4.1 onwards, the code analyzes the MPLS header, and if an IP header is available, it uses that for hash calculation.

If the MPLS label stack is more than four labels deep, the code stops looking for an IP header and reverts to the MAC header hash calculation.

Any Transport over MPLS

Any Transport over MPLS (AToM) transports Layer 2 packets over a Multiprotocol Label Switching (MPLS) backbone. This enables service providers to connect customer sites with existing Layer 2 networks by using a single, integrated, packet-based network infrastructure. Using this feature, service providers can deliver Layer 2 connections over an MPLS backbone, instead of using separate networks.

AToM encapsulates Layer 2 frames at the ingress PE router, and sends them to a corresponding PE router 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 a connection, called a *pseudowire*, between the routers. You specify this information on each PE router:

- The type of Layer 2 data that will be transported across the pseudowire, such as Ethernet, 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.

These topics describe the AToM feature:

- Control Word Processing

Control Word Processing

The control word contains forward explicit congestion notification (FECN), backward explicit congestion notification (BECN) and DE bits in case of frame relay connection.

Control word is mandatory for:

- ATM AAL5

L2VPN Nonstop Routing

The L2VPN Nonstop Routing (NSR) feature avoids label distribution path (LDP) sessions from flapping on events such as process failures (crash) and route processor failover (RP FO). NSR on process failure (crash) is supported by performing RP FO, if you have enabled NSR using NSR process failure switchover.

NSR enables the router (where failure has occurred) to maintain the control plane states without a graceful restart (GR). NSR, by definition, does not require any protocol extension and typically uses Stateful Switch Over (SSO) to maintain it's control plane states.

Traffic Injection from L2TPv3 over IPv6 Tunnel

Traffic Injection from L2TPv3 over IPv6 Tunnel feature allows you to inject diagnostic traffic through Layer 2 Tunneling Protocol version 3 (L2TPv3) Switched Port Analyzer (SPAN) tunnel. The diagnostic traffic allows you to monitor and troubleshoot the network traffic. You can send the diagnostic traffic from customer office (CO) using traffic generator towards the customer or towards the network.

In previous releases, with traffic mirroring feature the user was only able to send the mirrored traffic from customer towards the monitoring device, the mirror tunnel was unidirectional.

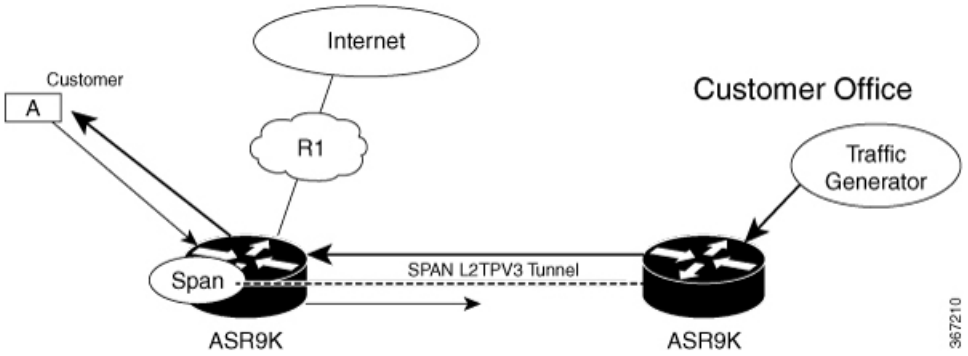
Traffic mirroring copies traffic from one or more source ports and sends the copied traffic to one or more destinations for analysis by a network analyzer or other monitoring device. Traffic mirroring does not affect the flow of traffic on the source interfaces or sub-interfaces, and allows the mirrored traffic to be sent to a destination interface or sub-interface

Restrictions

- This feature is not supported on bundle and sub-bundle interfaces, supported only on main and sub interfaces.
- Diagnostic traffic directed from network to customer does not traverse the same path as the actual non-diagnostic network to customer traffic. As a result, any issue with the core facing interface is not diagnosed.
- Diagnostic traffic will mix with the actual customer traffic. It is the responsibility of CO to ensure that it does not cause any problems to the customer, and CO is able to differentiate diagnostic traffic from customer traffic in the SPAN tunnel.
- Physical port features are not available in the inward inject path, so the problems in the physical port features, such as, EOAM or BIA MAC are not diagnosed.
- Bundle and other hash calculations, such as, ECMP, are not available, so the only customer interface supported is a physical interface.
- For Layer 3, only IPv4 and IPv6 diagnostic payload is supported.

Topology

Figure 9: Traffic Injection from L2TPv3 over IPv6 Tunnel



Consider a topology where an L2TPv3 tunnel is created between two ASR 9000 devices. Customer Office (CO) sends diagnostic traffic using traffic generator over L2TPv3 over IPv6 tunnel to the ASR 9000 router. The ASR 9000 router on the left-hand side sends the diagnostic traffic towards the customer as though it is sent from the network or sends the diagnostic traffic towards the network as though it is sent from the customer.

The diagnostic traffic header contains destination MAC address, source MAC address, and IP payload. If the header contains destination MAC address of the ASR 9000 router, the diagnostic traffic is sent to the network as though it sent from the customer. If the header contains source MAC address of the ASR 9000 router, the diagnostic traffic is sent to the customer as though it sent from the network.

Configure Traffic Injection from L2TPv3 over IPv6 Tunnel

Perform these tasks on ASR 9000 router, which is on the left-hand side to configure Traffic Injection from L2TPv3 over IPv6 Tunnel feature,

- Create a pseudowire monitor session with inject interface
- Attach the monitor session to an interface which needs to be spanned
- Configure L2VPN xconnect with monitor session

```
/* Create a pseudowire monitor session with inject interface */

Router# configure
Router(config)# monitor-session span1
Router(config-mon)# destination pseudowire
Router(config-mon)# inject-interface tenGigE 0/1/0/0/0
Router(config-mon)# commit
Router(config-mon)# end

/* Attach the monitor session to an interface which needs to be spanned */

Router# configure
Router(config)# int tenGigE 0/1/0/0/0
Router(config-subif)# monitor-session span1 ethernet
Router(config-if-mon)# commit

/* Configure L2VPN  xconnect with monitor session */

Router(config)# l2vpn
Router(config-l2vpn)# xconnect group xc-span1
Router(config-l2vpn-xc)# p2p span-session1
Router(config-l2vpn-xc-p2p)# monitor-session span1
Router(config-l2vpn-xc-p2p)# neighbor ipv6 1112::1:1 pw-id 101
Router(config-l2vpn-xc-p2p-pw)# pw-class ts
Router(config-l2vpn-xc-p2p-pw)# source 1111::1:1
Router(config-l2vpn-xc-p2p-pw)# l2tp static local cookie size 8 value 0x1 0xa1 local session
 101
Router(config-l2vpn-xc-p2p-pw)# l2tp static remote cookie size 8 value 0xa1 0x1 remote
session 101
Router(config-l2vpn-xc-p2p-pw)# commit
Router(config-l2vpn-xc-p2p-pw)# end
```

Running Configuration

```
configure
 monitor-session span1
  destination pseudowire
  inject-interface tenGigE 0/1/0/0/0
  !
!
```

```
configure
 int tenGigE 0/1/0/0/0
  monitor-session span1 ethernet
  !
!

l2vpn
 xconnect group xc-span1
  p2p span-session1
   monitor-session span1
   neighbor ipv6 1112::1:1 pw-id 101
    pw-class ts
    source 1111::1:1
    l2tp static
     local cookie size 8 value 0x1 0xa1
     local session 101
     remote cookie size 8 value 0xa1 0x1
     remote session 101
    !
   !
  !
```

Verification

Verify that the source MAC address and destination MAC address are matching.

```
/* Verify that the source MAC address is matching */

Router#show monitor-session span1 counters
Monitor-session span1
  TenGigE0/1/0/0/0.1
    Rx replicated: 248 packets, 247086 octets
    Tx replicated: 20001 packets, 20000094 octets
    Non-replicated: 0 packets, 0 octets


Router#show interface tenGigE 0/1/0/7/8 accounting
TenGigE0/1/0/7/8.1
  Protocol              Pkts In         Chars In     Pkts Out        Chars Out
  IPV6_UNICAST            10001         10480072        10005         10480632
  IPV6_MULTICAST              1              104            0                0
  IPV6_ND                     2              212            1               72


Router#show interface tenGigE 0/1/0/0/0 accounting
TenGigE0/1/0/0/0.1
  Protocol              Pkts In         Chars In     Pkts Out        Chars Out
  IPV6_UNICAST                2              136        10002          9780144
  IPV6_ND

/* Verify that the destination MAC address is matching */

Router#show monitor-session counters
Monitor-session span1
  TenGigE0/1/0/0/0.1
    Rx replicated: 10001 packets, 10000094 octets
    Tx replicated: 1 packets, 94 octets
    Non-replicated: 0 packets, 0 octets


Router#show interface tenGigE 0/1/0/7/8 accounting
TenGigE0/1/0/7/8.1
  Protocol              Pkts In         Chars In     Pkts Out        Chars Out
  IPV6_UNICAST            10000         10480000        10000         10480000
  IPV6_MULTICAST              0                0            1              104
  IPV6_ND                     0                0            1              104


Router#show interface tenGigE 0/1/0/0/0 accounting
TenGigE0/1/0/0/0.1
```

```
Protocol             Pkts In         Chars In     Pkts Out        Chars Out
IPV6_UNICAST           10000          9780000            0                0
IPV6_ND                    1               82            1               72
```

How to Implement Point to Point Layer 2 Services

This section describes the tasks required to implement Point to Point Layer 2 Services:

Configuring an Interface or Connection for Point to Point Layer 2 Services

Perform this task to configure an interface or a connection for Point to Point Layer 2 Services.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **l2transport**
4. **exit**
5. **interface** *type interface-path-id.subinterface* **l2transport**
6. **encapsulation dot1q** *vlan-id*
7. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters the Global Configuration mode.

Step 2 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/0
```

Enters interface configuration mode and configures an interface.

Step 3 **l2transport**

Example:

```
RP/0/RP0/CPU0:router(config-if)# l2transport
```

Enables L2 transport on the selected interface.

Step 4 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-if-l2)# exit
```

Exits the current configuration mode.

Step 5 **interface** *type interface-path-id.subinterface* **l2transport**

Example:

```
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/0.1 l2transport
```

Enters subinterface configuration mode and configures the subinterface as a layer 2 interface.

Step 6 **encapsulation dot1q** *vlan-id*

Example:

```
RP/0/RP0/CPU0:router(config-if)# encapsulation dot1q vln1
```

Assigns native VLAN ID to an interface trunking 802.1Q VLAN traffic.

Step 7 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuring Static Point-to-Point Cross-Connects

Note Consider this information about cross-connects when you configure static point-to-point cross-connects:

- An cross-connect is uniquely identified with the pair; the cross-connect name must be unique within a group.
- A segment (an attachment circuit or pseudowire) is unique and can belong only to a single cross-connect.
- A static VC local label is globally unique and can be used in one pseudowire only.
- No more than 16,000 cross-connects can be configured per router.

Note Static pseudowire connections do not use LDP for signaling.

Perform this task to configure static point-to-point cross-connects.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **xconnect group** *group-name*
4. **p2p** *xconnect-name*
5. **interface** *type interface-path-id*
6. **neighbor** *ip-address* **pw-id** *pseudowire-id*
7. **mpls static label local** { *value* } **remote** { *value* }
8. Use the **commit** or **end** command.
9. *show l2vpn xconnect group group name*

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters the Global Configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0/CPU0:router(config)# l2vpn
```

Enters L2VPN configuration mode.

Step 3 **xconnect group** *group-name*

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn)# xconnect group vlan_grp_1
```

Enters the name of the cross-connect group.

Step 4 **p2p** *xconnect-name*

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-xc)# p2p vlan1
```

Enters a name for the point-to-point cross-connect.

Step 5 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p)# interface TenGigE 0/0/0/0.1
```

Specifies the interface type and instance.

Step 6 **neighbor** *ip-address* **pw-id** *pseudowire-id*

Example:

```
RP/0/RP0/CPU0:routerr(config-l2vpn-xc-p2p)# neighbor 2.2.2.2 pw-id 2000
```

Configures the pseudowire segment for the cross-connect.

Optionally, you can disable the control word or set the transport-type to Ethernet or VLAN.

Step 7 **mpls static label local** { *value* } **remote** { *value* }

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p-pw)# mpls static label local 699 remote 890
```

Configures local and remote label ID values.

Step 8 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Step 9 *show l2vpn xconnect group group name*

Example:

```
RP/0/RP0/CPU0:show l2vpn xconnect group vlan_grp_1
```

Displays the name of the Point-to-Point cross-connect group you created.

Configuring Dynamic Point-to-Point Cross-Connects

Perform this task to configure dynamic point-to-point cross-connects.

Note For dynamic cross-connects, LDP must be up and running.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **xconnect group** *group-name*
4. **p2p** *xconnect-name*
5. **interworking ipv4**
6. **interface** *type interface-path-id*
7. **neighbor** *ip-address* **pw-id** *pseudowire-id*
8. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters the Global Configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0/CPU0:router(config)# l2vpn
```

Enters L2VPN configuration mode.

Step 3 **xconnect group** *group-name*

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn)# xconnect group grp_1
```

Enters the name of the cross-connect group.

Step 4 **p2p** *xconnect-name*

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-xc)# p2p vlan1
```

Enters a name for the point-to-point cross-connect.

Step 5 **interworking ipv4**

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-xc)# interworking ipv4
```

Configure the interworking for IPv4.

Step 6 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p)# interface GigabitEthernet0/0/0/0.1
```

Specifies the interface type ID. The choices are:

- GigabitEthernet: GigabitEthernet/IEEE 802.3 interfaces.
- TenGigE: TenGigabitEthernet/IEEE 802.3 interfaces.
- CEM: Circuit Emulation interface

Step 7 **neighbor** *ip-address* **pw-id** *pseudowire-id*

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-xc-p2p)# neighbor 2.2.2.2 pw-id 2000
```

Configures the pseudowire segment for the cross-connect.

Optionally, you can disable the control word or set the transport-type to Ethernet or VLAN.

Step 8 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuring Inter-AS

The Inter-AS configuration procedure is identical to the L2VPN cross-connect configuration tasks (see "Configuring Static Point-to-Point Cross-Connects" section and "Configuring Dynamic Point-to-Point Cross-Connects" section) except that the remote PE IP address used by the cross-connect configuration is now reachable through iBGP peering.

Note You must be knowledgeable about IBGP, EBGP, and ASBR terminology and configurations to complete this configuration.

Configuring L2VPN Quality of Service

This section describes how to configure L2VPN quality of service (QoS) in port mode and mode, VLAN mode, Frame Relay and ATM sub-interfaces.

Restrictions

The **l2transport** command cannot be used with any IP address, L3, or CDP configuration.

Configuring an L2VPN Quality of Service Policy in Port Mode

This procedure describes how to configure an L2VPN QoS policy in port mode.

Note In port mode, the interface name format does not include a subinterface number; for example, GigabitEthernet0/1/0/1.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id* **l2transport**
3. **service-policy** [**input** | **output**] [*policy-map-name*]
4. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters the configuration mode.

Step 2 **interface** *type interface-path-id* **l2transport**

Example:

```
RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0
```

Configures an interface or connection for L2 switching and specifies the interface attachment circuit.

Step 3 **service-policy** [**input** | **output**] [*policy-map-name*]

Example:

```
RP/0/RP0/CPU0:router(config-if)# service-policy input servpol1
```

Attaches a QoS policy to an input or output interface to be used as the service policy for that interface.

Step 4 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuring an L2VPN Quality of Service Policy in VLAN Mode

This procedure describes how to configure a L2VPN QoS policy in VLAN mode.

Note In VLAN mode, the interface name must include a subinterface. for example: GigabitEthernet0/1/0/1.1 and the l2transport command must follow the interface type on the same CLI line (for example: “interface GigabitEthernet 0/0/0/0.1 l2transport”).

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id.subinterface* **l2transport**
3. **service-policy** [**input** | **output**] [*policy-map-name*]
4. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:
```
RP/0/RP0/CPU0:router# configure
```

Enters the Global Configuration mode.

Step 2 **interface** *type interface-path-id.subinterface* **l2transport**

Example:

```
RP/0/RP0/CPU0:router(config)# interface GigabitEthernet0/0/0/0.1 l2transport
```

Configures an interface or connection for L2 switching.

Note In VLAN Mode, you must enter the **l2transport** keyword on the same line as the interface.

Step 3 **service-policy** [**input** | **output**] [*policy-map-name*]

Example:

```
RP/0/RP0/CPU0:router(config-if)# service-policy input servpol1
```

Attaches a QoS policy to an input or output interface to be used as the service policy for that interface.

Step 4 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuring Preferred Tunnel Path

This procedure describes how to configure a preferred tunnel path.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **pw-class** {*name*}
4. **encapsulation mpls**
5. **preferred-path** {**interface**} {**tunnel-ip** *value* | **tunnel-te** *value* | **tunnel-tp** *value*} [**fallback disable**]
6. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters the configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0/CPU0:router(config)# l2vpn
```

Enters L2VPN configuration mode.

Step 3 **pw-class** {*name*}

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn)# pw-class path1
```

Configures the pseudowire class name.

Step 4 **encapsulation mpls**

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls
```

Configures the pseudowire encapsulation to MPLS.

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

Example:

```
RP/0/RP0/CPU0:router(config-l2vpn-pwc-encap-mpls)# preferred-path interface tunnel-te 11 fallback
disable
```

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

Step 6 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Enabling Load Balancing with ECMP and FAT PW

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

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **pw-class** *{name}*
4. **encapsulation mpls**
5. **load-balancing flow-label both**
6. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:
```
RP/0/RP0/CPU0:router# configure
```
Enters the Global Configuration mode.

Step 2 **l2vpn**

Example:
```
RP/0/RP0/CPU0:router(config)# l2vpn
```
Enters the L2VPN configuration mode.

Step 3 **pw-class** *{name}*

Example:
```
RP/0/RP0/CPU0:router(config-l2vpn)# pw-class FAT_PW
```
Configures the pseudowire class name.

Step 4 **encapsulation mpls**

Example:
```
RP/0/RP0/CPU0:router(config-l2vpn-pwc)# encapsulation mpls
```
Configures the pseudowire encapsulation to MPLS.

Step 5 **load-balancing flow-label both**

Example:
```
RP/0/RP0/CPU0:router(config-l2vpn-pwc-encap-
mpls)# load-balancing flow-label both
```
Enables load-balancing on ECMPs. Also, enables the imposition and disposition of flow labels for the pseudowire.

Step 6 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuring L2VPN Nonstop Routing

Perform this task to configure L2VPN Nonstop Routing.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **nsr**
4. **logging nsr**
5. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:
```
RP/0/RP0/CPU0:router# configure
```

Enters Global Configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0/CPU0:router(config)# l2vpn
```

Enters the Global Configuration mode.

Step 3 **nsr**

Example:

```
RP/0/RP0/CPU0:router (config-l2vpn)# nsr
```

Enables L2VPN nonstop routing.

Step 4 **logging nsr**

Example:

```
RP/0/RP0/CPU0:router (config-l2vpn)# logging nsr
```

Enables logging of NSR events.

Step 5 Use the **commit** or **end** command.
commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuration Examples for Point to Point Layer 2 Services

In the following example, two traffic classes are created and their match criteria are defined. For the first traffic class called class1, ACL 101 is used as the match criterion. For the second traffic class called class2, ACL 102 is used as the match criterion. Packets are checked against the contents of these ACLs to determine if they belong to the class.

This section includes the following configuration examples:

L2VPN Interface Configuration: Example

This example shows how to configure an L2VPN interface:

```
configure
 interface TenGigE 0/0/0/0.1 l2transport
 encapsulation dot1q 1
 end
```

Point-to-Point Cross-connect Configuration: Examples

This section includes configuration examples for both static and dynamic p2p cross-connects.

Static Configuration

This example shows how to configure a static point-to-point cross-connect:

```
configure
 l2vpn
  xconnect group vlan_grp_1
   p2p vlan1
   interworking ipv4
   interface TenGigE 0/0/0/0.1
   neighbor 2.2.2.2 pw-id 2000
    mpls static label local 699 remote 890
    commit
```

Dynamic Configuration

This example shows how to configure a dynamic point-to-point cross-connect:

```
configure
  l2vpn
   xconnect group vlan_grp_1
    p2p vlan1
    interface TenGigE 0/0/0/0.1
    neighbor 2.2.1.1 pw-id 1
commit
```

Inter-AS: Example

This example shows how to set up an AC to AC cross-connect from AC1 to AC2:

```
router-id Loopback0

interface Loopback0
ipv4 address 127.0.0.1 255.255.255.0
```

```
!
interface GigabitEthernet0/1/0/0.1 l2transport encapsulation dot1q 1
!
interface GigabitEthernet0/0/0/3
ipv4 address 127.0.0.1 255.255.255.0
keepalive disable
!
interface GigabitEthernet0/0/0/4
ipv4 address 127.0.0.1 255.255.255.0
keepalive disable
!
router ospf 100
log adjacency changes detail
area 0
interface Loopback0
!
interface GigabitEthernet0/0/0/3
!
interface GigabitEthernet0/0/0/4
!
!
!
router bgp 100
address-family ipv4 unicast
allocate-label all
!
neighbor 40.0.0.5
remote-as 100
update-source Loopback0
address-family ipv4 unicast
!
address-family ipv4 labeled-unicast
!
!
!
l2vpn
xconnect group xc1
p2p ac2ac1
interface GigabitEthernet0/1/0/0.1
neighbor 20.0.0.5 pw-id 101
!
p2p ac2ac2
interface GigabitEthernet0/1/0/0.2
neighbor 20.0.0.5 pw-id 102
!
p2p ac2ac3
interface GigabitEthernet0/1/0/0.3
neighbor 20.0.0.5 pw-id 103
!
p2p ac2ac4
interface GigabitEthernet0/1/0/0.4
neighbor 20.0.0.5 pw-id 104
!
p2p ac2ac5
interface GigabitEthernet0/1/0/0.5
neighbor 20.0.0.5 pw-id 105
!
p2p ac2ac6
interface GigabitEthernet0/1/0/0.6
neighbor 20.0.0.5 pw-id 106
!
p2p ac2ac7
interface GigabitEthernet0/1/0/0.7
neighbor 20.0.0.5 pw-id 107
!
p2p ac2ac8
interface GigabitEthernet0/1/0/0.8
neighbor 20.0.0.5 pw-id 108
!
p2p ac2ac9
interface GigabitEthernet0/1/0/0.9
neighbor 20.0.0.5 pw-id 109
!
```

```
p2p ac2ac10
interface GigabitEthernet0/1/0/0.10
neighbor 20.0.0.5 pw-id 110
!
!
!
mpls ldp
router-id Loopback0
log
neighbor
!
interface GigabitEthernet0/0/0/3
!
interface GigabitEthernet0/0/0/4
!
!
end
```

L2VPN Quality of Service: Example

This example shows how to attach a service-policy to an L2 interface in port mode:

```
configure
  interface GigabitEthernet 0/0/0/0
  l2transport

commit
```

Preferred Path: Example

This example shows how to configure preferred tunnel path:

```
configure
l2vpn
pw-class path1
encapsulation mpls
preferred-path interface tunnel-ip value  fallback disable
```

Enabling Load Balancing with FAT PW: Example

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

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

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

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

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

```
l2vpn
bridge group group1
    bridge-domain domain1
        neighbor 1.1.1.1 pw-id 1
             pw-class class1
        !
        vfi vfi-manual
            neighbor 2.2.2.2 pw-id 2
                pw-class class2
            !
        !
    !
    bridge-domain domain2
        vfi vfi2-auto-bgp
                autodiscovery bgp
                   signaling-protocol bgp
                       load-balancing flow-label both static
                !
            !
        !
    !
    bridge-domain domain3
        vfi vfi2-auto-ldp
                autodiscovery bgp
                   signaling-protocol ldp
                       load-balancing flow-label both static
                !
            !
        !
    !
!
```

AToM Cross Connect Configuration: Example

This section includes configuration examples for all supported AToM Cross Connects.

```
l2vpn
pseudowire-class ipiw
  encapsulation mpls
!
xconnect group port
  p2p port1
    interface GigabitEthernet0/0/0/2
    neighbor 11.11.11.11 pw-id 300 pw-class ipiw
  !
!
xconnect group vlan
  p2p vlan1
    interface GigabitEthernet0/0/0/3.1
    neighbor 11.11.11.11 pw-id 400 pw-class ipiw
  !
!
xconnect group frame-relay
  p2p frame1
    interface POS0/2/0/1.20
    neighbor 11.11.11.11 pw-id 600 pw-class ipiw
  !
!
xconnect group atm
  p2p atm1
    interface ATM0/3/0/1.200
    neighbor 11.11.11.11 pw-id 700 pw-class ipiw
```

```
  !
  p2p atm2
    interface ATM0/3/0/1.300
    neighbor 11.11.11.11 pw-id 800 pw-class ipiw
```

Configuring L2VPN over GRE Tunnels: Example

The following example shows how to configure L2VPN over GRE tunnels:

```
interface tunnel-ip101
 ipv4 address 150.10.1.204 255.255.255.0
 ipv6 address 150:10:1::204/64
 tunnel mode gre ipv4
 tunnel source Loopback1
 tunnel destination 100.1.1.202

router ospf 1
 router-id 100.0.1.204
 cost 1
 router-id Loopback0
 area 1
  interface Loopback0
  !
  interface tunnel-ip101

mpls ldp
 router-id 100.0.1.204
 interface tunnel-ip101

l2vpn
 xconnect group pe2
  p2p 2001
   interface GigabitEthernet0/2/0/0.2001
   neighbor 100.0.1.202 pw-id 2001
```

Configuring L2VPN Nonstop Routing: Example

This example shows how to configure L2VPN Nonstop Routing.

```
config
l2vpn
 nsr
 logging nsr
```