Configuring Point to Point Layer 2 Services

This chapter provides conceptual and configuration information for point-to-point Layer 2 (L2) connectivity on Cisco NCS 4000 Series routers.

Layer 2 Virtual Private Network Overview

Layer 2 Virtual Private Network (L2VPN) emulates the behavior of a LAN across an L2 switched, IP or MPLS-enabled IP network, allowing Ethernet devices to communicate with each other as they would when connected to a common LAN segment. Point-to-point L2 connections are vital when creating L2VPNs.

As Internet service providers (ISPs) look to replace their Frame Relay or Asynchronous Transfer Mode (ATM) infrastructures with an IP infrastructure, there is a need to provide standard methods of using an L2 switched, IP or MPLS-enabled IP infrastructure. These methods provide a serviceable L2 interface to customers; specifically, 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 these 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 TE, Flex LSP).
- L2VPN process databases include all information related to circuits and their connections.

Ethernet Virtual Circuit

Ethernet virtual circuits (EVCs) define a Layer 2 bridging architecture that supports Ethernet services. An EVC is defined by the Metro-Ethernet Forum (MEF) as an association between two or more user network interfaces that identifies a point-to-point or multipoint-to-multipoint path within the service provider network. An EVC is a conceptual service pipe within the service provider network. On Cisco NCS 4000 Series Routers, the EVC is implemented as a pseudowire (PW). This section explains the basic rules for configuring EVC:

- **Enable L2 transport on an interface (l2transport command)** - A packet must be received on an interface configured with the l2transport keyword in order to be processed by the L2VPN feature. This interface can be a main interface, where the l2transport command is configured under the interface config mode, or a subinterface, where the l2transport keyword is configured after the sub interface number.

  Example:

  ```
  interface TenGigE0/0/0/2 l2transport
  ```

  ```
  interface TenGigE0/6/0/6.11 l2transport
  ```

- **Incoming Interface Matching (encapsulation command)** - This command is used to specify matching criteria. A longest match lookup determines the incoming interface of the packet. The longest match lookup checks these conditions in this order to match the incoming packet to a subinterface:
  - The incoming frame has two dot1q tags and matches a subinterface configured with the same two dot1q tags (802.1Q tunneling, or QinQ). This is the longest possible match.
  - The incoming frame has two dot1q tags and matches a subinterface configured with the same dot1q first tag and any for the second tag.
  - The incoming frame has one dot1q tag and matches a subinterface configured with the same dot1q tag and the exact keyword.
  - The incoming frame has one or more dot1q tags and matches a subinterface configured with one of the dot1q tags.
  - The incoming frame has no dot1q tags and matches a subinterface configured with the **encapsulation untagged** command.
  - The incoming frame fails to match any other subinterface, so it matches a subinterface configured with the **encapsulation default** command.

Note Assignment of incoming frames to a subinterface based on source MAC adress is not supported.

Following **examples** explain the use of the encapsulation command:

1. To match any tagged or untagged traffic that has not been matched by another subinterface with a longest match:

```
interface TenGigE0/1/0/3.1 l2transport
 encapsulation default
```

2. When there are multiple subinterfaces, run the longest match test on the incoming frame in order to determine the incoming interface:

```
interface TenGigE0/1/0/3.1 l2transport
 encapsulation default
!
interface TenGigE0/1/0/3.2 l2transport
 encapsulation dot1q 2
!
interface TenGigE0/1/0/3.3 l2transport
 encapsulation dot1q 2 second-dot1q 3
```

Note

- A QinQ frame with an outer VLAN tag 2 and an inner VLAN tag 3 could match the .1, .2, or .3 subinterfaces but it is assigned to the .3 subinterface because of the longest match rule. Two tags on .3 are longer than one tag on .2 and longer than no tags on .1.
- A QinQ frame with an outer VLAN tag 2 and an inner VLAN tag 4 is assigned to the .2 subinterface because encapsulation dot1q 2 can match dot1q frames with just the VLAN tag 2 but can also match QinQ frames with an outer tag 2. Refer to Example 3(the exact keyword) if you do not want to match the QinQ frames.
- A QinQ frame with an outer VLAN tag 3 matches the .1 subinterface.
- A dot1q frame with a VLAN tag 2 matches the .2 subinterface.
- A dot1q frame with a VLAN tag 3 matches the .1 subinterface.

3. To match a dot1q frame and not a QinQ frame, use the exact keyword:

```
interface TenGigE0/1/0/3.2 l2transport
 encapsulation dot1q 2 exact
```

Note This configuration does not match QinQ frames with an outer VLAN tag 2 because it matches only frames with exactly one VLAN tag.

4. Use the untagged keyword in order to match only untagged frames :

```
interface TenGigE0/1/0/3.1 l2transport
 encapsulation default
```

```
!
interface TenGigE0/1/0/3.2 l2transport
 encapsulation untagged
!
interface TenGigE0/1/0/3.3 l2transport
 encapsulation dot1q 3
```

Note

- Dot1q frames with a VLAN tag 3 or QinQ frames with an outer tag 3 match the .3 subinterfaces.
- All other dot1q or QinQ frames match the .1 subinterface.
- Frames without a VLAN tag match the .2 subinterface.

5. The "any" keyword can be used as wildcard:

```
interface TenGigE0/1/0/3.4 l2transport
 encapsulation dot1q 4 second-dot1q any
!
interface TenGigE0/1/0/3.5 l2transport
 encapsulation dot1q 4 second-dot1q 5
```

Note

- Both subinterfaces .4 and .5 could match QinQ frames with tags 4 and 5, but the frames are assigned to the .5 subinterfaces because it is more specific. This is the longest match rule.
- The "any" keyword option is not applicable with single VLAN dot1q or dot1ad. For example following is not supported:

```
encapsulation dot1q any
or
encapsulation dot1ad any
```

6. Ranges of VLAN tags can be used:

```
interface TenGigE0/1/0/3.6 l2transport
 encapsulation dot1q 6-10
```

Note

- Per line card maximum 32 dot1q or dot1ad ranges (including both inner or outer range) can be configured.
- Multiple VLAN tag values or ranges are not supported.

7. The encapsulation dot1q second-dot1q command uses the Ethertype 0x8100 for the outer and inner tags because this is the Cisco method to encapsulate QinQ frames. According to IEEE, however, the Ethertype 0x8100 should be reserved for 802.1q frames with one VLAN tag, and an outer tag with Ethertype 0x88a8 should be used for QinQ frames. The outer tag with Ethertype 0x88a8 can be configured with the dot1ad keyword:

```
interface TenGigE0/1/0/3.12 l2transport
 encapsulation dot1ad 12 dot1q 100
```

8. In order to use the old Ethertype 0x9100 or 0x9200 for the QinQ outer tags, use the dot1q tunneling ethertype command under the main interface of the QinQ subinterface:

```
interface TenGigE0/1/0/3
 dot1q tunneling ethertype [0x9100|0x9200]
!
interface TenGigE0/1/0/3.13 l2transport
 encapsulation dot1q 13 second-dot1q 100
```

Note

- The outer tag has an Ethertype of 0x9100 or 0x9200, and the inner tag has the dot1q Ethertype 0x8100.
- Per interface only two Ethertype are supported. Whenever custom Ethertype are added for an interface, dot1ad configuration should not be present on that interface.

- **VLAN Manipulation (rewrite command)** - On a Cisco NCS 4000 Router that uses the EVC infrastructure, the default action is to preserve the VLAN tags on the incoming frame. But, the EVC infrastructure allows you to manipulate the tags with the rewrite command. Use the rewrite command to modify the default , so you can pop (remove), translate, or push (add) tags to the incoming VLAN tag stack.

Note Egress vlan filter is not supported, so when packet is egressing no egress vlan checks are performed.

Following **examples** explain the use of the rewrite command:

- The pop keyword lets you remove a QinQ tag from an incoming dot1q frame. This example removes the outer tag 13 of the incoming QinQ frame and forwards the frame with the dot1q tag 100 on top:

```
interface TenGigE0/1/0/3.13 l2transport
 encapsulation dot1q 13 second-dot1q 100
 rewrite ingress tag pop 1 symmetric
```

Note The behavior is always symmetric, which means that the outer tag 13 is popped in the ingress direction and pushed in the egress direction.

- The translate keyword lets you replace one or two incoming tags by one or two new tags:

```
RP/0/RP0:hostname(config-subif)#interface TenGigE0/1/0/3.3
   l2transport
RP/0/RP0:hostname(config-subif)# encapsulation dot1q 3
RP/0/RP0:hostname(config-subif)#rewrite ingress tag translate ?
  1-to-1  Replace the outermost tag with another tag
  2-to-2  Replace the outermost two tags with two other tags
RP/0/RP0:hostname(config-subif)#rewrite ingress tag translate 1-to-1 ?
  dot1ad  Push a Dot1ad tag
  dot1q   Push a Dot1Q tag
RP/0/RP0:hostname(config-subif)#rewrite ingress tag translate 1-to-1
   dot1q 4
```

```
RP/0/RP0:hostname(config-subif)#show config
Building configuration...
!! IOS XR Configuration 4.3.0
interface TenGigE0/1/0/3.3 l2transport
 encapsulation dot1q 3
 rewrite ingress tag translate 1-to-1 dot1ad 20 symmetric
!
end
```

Note

- The symmetric keyword is added automatically because it is the only supported mode.
- translate 1-to-2 and 2-to-1 are not supported.
- translate 1-to-1 is not supported with dot1ad option.
- translate 2-to-2 is not supported for VPWS on NCS4K-2H10T-OP-KS card.

- The push keyword lets you add a QinQ tag to an incoming dot1q frame:

```
interface TenGigE0/1/0/3.4 l2transport
 encapsulation dot1q 4
 rewrite ingress tag push dot1q 100 symmetric
```

Note

- An outer QinQ tag 100 is added to the incoming frame with a dot1q tag 4. In the egress direction, the QinQ tag is popped.
- With encapsulation default, vlan push operations are not supported.
- Adding two tags with "push" is supported only with singled tagged or untagged encapsulation like for example:

  ```
  rewrite ingress tag push dot1q 20 second-dot1q 200 symmetric
  ```

- Adding two tags with "push" is not supported for VPWS on NCS4K-2H10T-OP-KS card.

Ethernet Wire Service

An Ethernet Wire Service is a service that emulates a point-to-point Ethernet segment. This is similar to Ethernet private line (EPL), a Layer 1 point-to-point service, except the provider edge operates at Layer 2 and typically runs over a Layer 2 network. The EWS encapsulates all frames that are received on a particular UNI and transports these frames to a single-egress UNI without reference to the contents contained within the frame. The operation of this service means that an EWS can be used with VLAN-tagged frames. The VLAN tags are transparent to the EWS (bridge protocol data units [BPDUs])-with some exceptions. These exceptions include IEEE 802.1x, IEEE 802.2ad, and IEEE 802.3x, because these frames have local significance and it benefits both the customer and the Service Provider to terminate them locally.

The customer side has these types:

- Untagged
- Single tagged
- Double tagged
- 802.1q
- 802.1ad

E-Line Service

E-Line service provides a point-to-point EVC between two UNIs. There are two types of E-Line services:

- Ethernet Private Line (EPL)
  - No service multiplexing allowed
  - Transparent
  - No coordination between customer and SP on VLAN ID map
- Ethernet Virtual Private Line (EVPL)
  - Allows service multiplexing
  - No need for full transparency of service frames

EPL and EVPL services are provided through:

- Layer 2 Local Switching, on page 7
- VPWS, on page 13

Layer 2 Local Switching

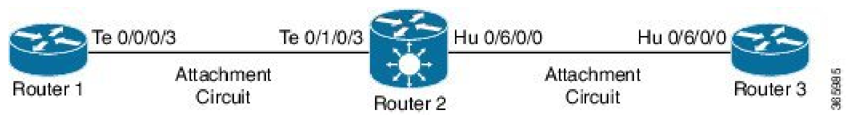
Local switching is a point-to-point circuit internal to a single Cisco NCS 4000 Series router, also known as local connect. Local switching allows you to switch L2 data between two interfaces of the same type, (for example, Ethernet to Ethernet) and on the same router. The interfaces can be on the same line card, or on two different line cards. During these types of switching, Layer 2 address is used instead of the Layer 3 address.

A local switching connection switches L2 traffic from one attachment circuit (AC) to the other. The two ports configured in a local switching connection are ACs with respect to that local connection.

Main Interface

The basic topology is a local cross connect between two main interfaces:

Figure 1:



Router2 takes all traffic received on Te 0/1/0/3 and forwards it to Hu 0/6/0/0 and vice versa.

While router1 and router3 appear to have a direct back-to-back cable in this topology, this is not the case because router2 is actually translating between the TenGigE and HundredGigE interfaces. Router2 can run features on these two interfaces.

A basic point-to-point cross connect is configured between two main interfaces that are configured as l2transport on router2:

```
 interface TenGigE0/1/0/3 l2transport
 !
!
interface HundredGigE0/6/0/0
 l2transport
 !
!
l2vpn
 xconnect group test
  p2p p2p1
   interface HundredGigE0/6/0/0
   interface TenGigE0/1/0/3
  !
```

On router1 and router3, the main interfaces are configured with IPv4 address:

```
RP/0/RP0:router1#sh run int Te 0/0/0/3
interface TenGigE0/0/0/3

 ipv4 address 10.1.1.1 255.255.255.0
!

RP/0/RP0:router1#ping 10.1.1.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 2/8/32 ms
```

Router1 sees router3 as a neighbor and can ping 10.1.1.2 (the interface address of router3) as if the two routers were directly connected.

Because there is no subinterface configured on router2, incoming frames with a VLAN tag are transported transparently when dot1q subinterfaces are configured on router1 and router3:

```
RP/0/RP0:router1#sh run int Te 0/0/0/3.2
interface TenGigE0/0/0/3.2
 ipv4 address 10.1.2.1 255.255.255.0
 dot1q vlan 2
!

RP/0/RP0:router1#ping 10.1.2.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 2/3/5 ms
```

After 10,000 pings from router1 to router3, you can use the show interface and show l2vpn commands in order to ensure that ping requests received by router2 on one AC are forwarded on the other AC and that ping replies are handled the same way in reverse.

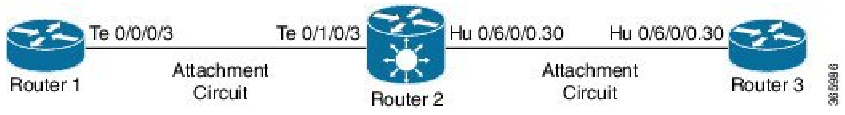
```
RP/0/RP0:router2#sh int Te 0/1/0/3
  TenGigE0/0/0/3 is up, line protocol is up
  Interface state transitions: 1
  Hardware is TenGigE, address is 0024.986c.63f1 (bia 0024.986c.63f1)
  Description: static lab connection to acdc 0/0/0/3 - dont change
  Layer 2 Transport Mode
  MTU 1514 bytes, BW 1000000 Kbit (Max: 1000000 Kbit)
```

```
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation ARPA,
  Full-duplex, 1000Mb/s, SXFD, link type is force-up
  output flow control is off, input flow control is off
  loopback not set,
  Last input 00:00:00, output 00:00:00
  Last clearing of "show interface" counters 00:01:07
  5 minute input rate 28000 bits/sec, 32 packets/sec
  5 minute output rate 28000 bits/sec, 32 packets/sec
     10006 packets input, 1140592 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 6 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     10007 packets output, 1140832 bytes, 0 total output drops
     Output 0 broadcast packets, 7 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions


RP/0/RP0:router2#sh int Hu 0/6/0/0
HundredGigE0/6/0/0 is up, line protocol is up
  Interface state transitions: 3
  Hardware is HundredGigE, address is 0024.98ea.038b (bia 0024.98ea.038b)
  Layer 1 Transport Mode is LAN
  Description: static lab connection to putin 0/6/0/0 - dont change
  Layer 2 Transport Mode
  MTU 1514 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation ARPA,
  Full-duplex, 10000Mb/s, LR, link type is force-up
  output flow control is off, input flow control is off
  loopback not set,
  Last input 00:00:00, output 00:00:06
  Last clearing of "show interface" counters 00:01:15
  5 minute input rate 27000 bits/sec, 30 packets/sec
  5 minute output rate 27000 bits/sec, 30 packets/sec
     10008 packets input, 1140908 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 8 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     10006 packets output, 1140592 bytes, 0 total output drops
     Output 0 broadcast packets, 6 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions


RP/0/RP0:router2#sh l2vpn xconnect group test
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
        SB = Standby, SR = Standby Ready, (PP) = Partially Programmed

XConnect                Segment 1                   Segment 2
Group      Name    ST   Description         ST      Description         ST
---------------------   --------------------------  --------------------------
test       p2p1    UP   Hu0/6/0/0           UP      Te0/1/0/3           UP
--------------------------------------------------------------------------------
RP/0/RP0:router2#sh l2vpn xconnect group test det

Group test, XC p2p1, state is up; Interworking none
  AC: TenGigE0/0/0/3, state is up
    Type Ethernet
    MTU 1500; XC ID 0x1080001; interworking none
    Statistics:
      packets: received 10008, sent 10006
      bytes: received 1140908, sent 1140592
  AC: TenGigE0/1/0/3, state is up
    Type Ethernet
    MTU 1500; XC ID 0x1880003; interworking none
    Statistics:
      packets: received 10006, sent 10008
      bytes: received 1140592, sent 1140908

RP/0/RP0#sh l2vpn forwarding interface TenGigE 0/0/0/10 hardware ingress detail location
0/RP0
Local interface: TenGigE0/0/0/10, Xconnect id: 0x3a, Status: up
  Segment 1
    AC, TenGigE0/0/0/10, Ethernet port mode, status: Bound
    Statistics:
      packets: received 777274547, sent 731226431
      bytes: received 99047365649, sent 93179272680
      packets dropped: PLU 0, tail 0
      bytes dropped: PLU 0, tail 0
  Segment 2
    AC, TenGigE0/1/0/0/100, Ethernet port mode, status: Bound

RP/0/RP0:router2#sh l2vpn forwarding interface Hu 0/6/0/0 hardware egress
   detail location 0/0
Local interface: HundredGigE0/6/0/0, Xconnect id: 0x1080001, Status: up
  Segment 1
    AC, HundredGigE0/6/0/0, Ethernet port mode, status: Bound
    Statistics:
      packets: received 10028, sent 10027
      bytes: received 1143016, sent 1142732
      packets dropped: PLU 0, tail 0
      bytes dropped: PLU 0, tail 0
  Segment 2
    AC, TenGigE0/1/0/3, Ethernet port mode, status: Bound

  Platform AC context:
  Egress AC: Local Switch, State: Bound
    Flags: Remote is Simple AC
  XID: 0x00000001, SHG: None
  Ingress uIDB: 0x0007, Egress uIDB: 0x0007, NP: 0, Port Learn Key: 0
  NP0
    Egress uIDB:
      Flags: L2, Status, Done
      Stats ptr: 0x000000
      VPLS SHG: None
      VLAN1: 0, VLAN1 etype: 0x0000, VLAN2: 0, VLAN2 etype: 0x0000
      UIDB IF Handle: 0x04000240, Search VLAN Vector: 0
      QOS ID: 0, QOS format: 0
    Xconnect ID: 0x00000001, NP: 0
      Type: AC, Remote type: AC
      Flags: Learn enable
      uIDB Index: 0x0007, LAG pointer: 0x0000
      Split Horizon Group: None
```

Subinterfaces and VLAN Manipulation

The basic topology is a local cross connect between a main interface and a sub interface:

Figure 2:



Following section describes how flexible rewrite capabilities give multiple ways to manipulate the VLAN :

1. Main Interface and Dot1q Subinterface

   In this example, the main interface is on one side, and the dot1q subinterface is on the other side:

   This is the main interface on router1:

```
RP/0/RP0:router1#sh run int te 0/0/0/3
interface TenGigE0/0/0/3
 description static lab connection to router2 0/1/0/3
 ipv4 address 10.1.1.1 255.255.255.0
!
```

   This is the dot1q subinterface on router2:

```
RP/0/RP0:router2#sh run int te 0/1/0/3
interface TenGigE0/1/0/3
 description static lab connection to router1 0/0/0/3
 l2transport

RP/0/RP0:router2#sh run int hu 0/6/0/0.30
interface HundredGigE0/6/0/0.30 l2transport
 encapsulation dot1q 2
 rewrite ingress tag pop 1 symmetric

RP/0/RP0:router2#sh run l2vpn xconnect group test
l2vpn
 xconnect group test
  p2p p2p2
   interface HundredGigE0/6/0/0.30
   interface TenGigE0/1/0/3
```

   There is now an l2transport keyword in the subinterface name of HundredGigE0/6/0/0.30. Router3 sends dot1q frames with tag 2, which match the HundredGigE0/6/0/0.30 subinterface on router2.

   The incoming tag 2 is removed in the ingress direction by the rewrite ingress tag pop 1 symmetric command. Since the tag has been removed in the ingress direction on the HundredGigE0/6/0/0.30, the packets are sent untagged in the egress direction on TenGigE0/1/0/3.

   Router1 sends untagged frames, which match the main interface TenGigE0/1/0/3.

   There is no rewrite command on TenGigE0/1/0/3, so no tag is popped, pushed, or translated.

   When packets have to be forwarded out of HundredGigE0/6/0/0.30, the dot1q tag 2 is pushed due to the symmetric keyword in the rewrite ingress tag pop 1 command. The command pops one tag in the ingress direction but symmetrically pushes one tag in the egress direction. This is an example on router3:

```
RP/0/RP0:router3#sh run int hu 0/6/0/0.30
interface HundredGigE0/6/0/0.30
 ipv4 address 10.1.1.2 255.255.255.0
 encapsulation dot1q 2
```

   Monitor the subinterface counters with the same show interface and show l2vpn commands:

```
RP/0/RP0:router2#clear counters
Clear "show interface" counters on all interfaces [confirm]
RP/0/RP0:router2#clear l2vpn forwarding counters
```

```
RP/0/RP0:router2#
RP/0/RP0:router2#
RP/0/RP0:router2#sh int HundredGigE0/6/0/0.30
HundredGigE0/6/0/0.30 is up, line protocol is up
  Interface state transitions: 1
  Hardware is VLAN sub-interface(s), address is 0024.98ea.038b
  Layer 2 Transport Mode
  MTU 1518 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
     reliability Unknown, txload Unknown, rxload Unknown
  Encapsulation 802.1Q Virtual LAN,
    Outer Match: Dot1Q VLAN 2
    Ethertype Any, MAC Match src any, dest any
  loopback not set,
  Last input 00:00:00, output 00:00:00
  Last clearing of "show interface" counters 00:00:27
     1000 packets input, 122000 bytes
     0 input drops, 0 queue drops, 0 input errors
     1002 packets output, 122326 bytes
     0 output drops, 0 queue drops, 0 output errors


RP/0/RP0:router2#sh l2vpn xconnect detail

Group test, XC p2p2, state is up; Interworking none
  AC: HundredGigE0/6/0/0.30, state is up
    Type VLAN; Num Ranges: 1
    VLAN ranges: [2, 2]
    MTU 1500; XC ID 0x1080001; interworking none
    Statistics:
      packets: received 1001, sent 1002
      bytes: received 118080, sent 118318
      drops: illegal VLAN 0, illegal length 0
  AC: TenGigE0/1/0/3, state is up
    Type Ethernet
    MTU 1500; XC ID 0x1880003; interworking none
    Statistics:
      packets: received 1002, sent 1001
      bytes: received 114310, sent 114076
```

As expected, the number of packets received on HundredGigE0/6/0/0.30 matches the number of packets sent on TenGigE0/1/0/3 and vice versa.

2. Subinterface with Encapsulation

   Instead of the main interface on TenGigE0/1/0/3, you can use a subinterface with encapsulation default in order to catch all frames or with encapsulation untagged in order to match only untagged frames:

```
RP/0/RP0:router2#sh run interface TenGigE0/1/0/3.1
interface TenGigE0/1/0/3.1 l2transport
 encapsulation untagged

RP/0/RP0:router2#sh run int HundredGigE0/6/0/0.30
interface HundredGigE0/6/0/0.30 l2transport
 encapsulation dot1q 2
 rewrite ingress tag pop 1 symmetric

RP/0/RP0:router2#sh run l2vpn xconnect group test
l2vpn
 xconnect group test
  p2p p2p3
   interface HundredGigE0/6/0/0.30
   interface TenGigE0/1/0/3.1
```

3. Ingress Direction on TenGigE0/1/0/3.1

Rather than pop tag 2 in the ingress direction on HundredGigE0/6/0/0.30, you can push tag 2 in the ingress direction on TenGigE0/1/0/3.1 and not do anything on HundredGigE0/6/0/0.30:

```
RP/0/RP0:router2#sh run int  HundredGigE0/6/0/0.30
interface HundredGigE0/6/0/0.30 l2transport
 encapsulation dot1q 2

RP/0/RP0:router2#sh run interface TenGigE0/1/0/3.1
interface TenGigE0/1/0/3.1 l2transport
 encapsulation untagged
 rewrite ingress tag push dot1q 2 symmetric

RP/0/RP0:router2#sh run int HundredGigE0/6/0/0.30
interface HundredGigE0/6/0/0.30 l2transport
 encapsulation dot1q 2

RP/0/RP0:router2#sh run l2vpn xconnect group test
l2vpn
 xconnect group test
  p2p p2p3
   interface HundredGigE0/6/0/0.30
   interface TenGigE0/1/0/3.1
```

Thus, you can see that the EVC model with the encapsulation and rewrite commands gives you great flexibility to match and manipulate VLAN tags.

Limitations :

- Pseudo wire redundancy is not supported

VPWS

Virtual Private Wire Services (VPWS), also known as Ethernet-over-MPLS (EoMPLS), allow two L2VPN Provider Edge (PE) devices to tunnel the 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 cloud. The two L2VPN PEs are typically connected at two different sites with an MPLS core between them. The two attachment circuits (ACs)connected at each L2VPN PE are linked by a pseudo wire (PW) over the MPLS network, which is the MPLS PW. The pseudo wire is a virtual point-to-point circuit and is always a type 5 virtual connection (VC). Type 4 VCs and Control Word (CW) are not supported.

For more information on pseudo wire types, see Type 5 Pseudo Wires, on page 18. The number of PWs supported on NCS4K-4H-OPW-QC2 and NCS4K-2H10T-OP-KS cards is 1000.

The two PEs establish an MPLS LDP targeted session between themselves so they can establish and control the status of the PW. An MPLS LDP targeted session is a label distribution session between routers that are not directly connected. When you create an MPLS traffic engineering tunnel interface, you need to establish a label distribution session between the tunnel headend and the tailend routers.The MPLS LDP targeted session is established over:

- Flex LSP. For more information
- MPLS TE.

EoMPLS features are described in this subsection:

- Ethernet Port Mode, on page 17

Pseudo wire redundancy is not supported.

VPWS Scale

The following table displays the scale numbers for VPWS:

Line Card	Scale
NCS4K-2H10T-OP-KS line card	1000
NCS4K-4H-OPW-QC2 line card	1000

Sytem	Scale
Node with NCS4K-2H10T-OP-KS and NCS4K-4H-OPW-QC2 line cards	4000
Node with NCS4K-4H-OPW-QC2 line cards	4000

FAT Pseudo Wires

In a VPWS network, the flow aware transport (FAT) of pseudowires can be used for load balancing traffic across LDP-signaled pseudowires. A flow label is a unique identifier to distinguish a flow within the pseudowire. These flow labels enable load balancing of MPLS packets across equal cost multipath (ECMP) paths or link aggregation groups (LAGs).When the pseudowire is configured to use the flow labels for load balancing, packets arriving at the ingress PE node are processed. A flow label is inserted for each packet between the VC label and control word.The flow label is derived from the payload of the inbound packet using a hash-key algorithm. The ingress router pushes the flow label to the label stack of the packet. At the egress PE node, hashing is performed using the terminated headers, including the flow label to balance traffic across LAG members.

To balance the load based on flow labels, use the **load-balancing flow-label**command in the l2vpn pseudowire class mpls configuration submode.

Use the **fat-pw load-balance terminated** command to configure the ingress interface of the egress PE node so that LAG hashing is performed using the terminating header of the traffic that is received.

Pseudowire Call Admission Control (CAC)

You can use the Pseudowire Call Admission Control (PW CAC) process to check for bandwidth constraints and ensure that after the path is signaled, the links (pseudowires participating in the bidirectional LSP association have the required bandwidth. Only pseudowires with sufficient bandwidth are admitted in the bidirectional LSP association process. The PW CAC feature works only when the PW is configured with a L2VPN preferred path tunnel.

You can configure bandwidth allocation and call admission control on layer 2 circuits. When you configure bandwidth on a layer 2 circuit, attempts to establish a bidirectional LSP is preceded by a check of the available bandwidth on the network. The available bandwidth is compared to the bandwidth requested by the LSP. If there is insufficient bandwidth, the circuit is not established.

To verify if the requested bandwidth has been allocated and whether the PW is up, use the **l2vpn xconnect detail** command. The following examples display the verification output.

Example:1

Requested bandwidth is available. In this scenario, the PW is up.

```
Group VPWS, XC p1, state is up; Interworking none
  AC: FortyGigE0/9/0/9.1, state is up
    Type VLAN; Num Ranges: 0
    MTU 9202; XC ID 0x1; interworking none
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0
      drops: illegal VLAN 0, illegal length 0
  PW: neighbor 3.3.3.3, PW ID 1, state is up ( established )
    PW class vpws1, XC ID 0xc0000001
    Encapsulation MPLS, protocol LDP
    Source address 1.1.1.1
    PW type Ethernet, control word disabled, interworking none
    PW backup disable delay 0 sec
    Sequencing not set
    Preferred path tunnel TE 1, fallback enabled
    Required BW = 1000 Admited BW = 1000
    PW Status TLV in use
      MPLS         Local                          Remote
      ------------ ------------------------------ -----------------------------
      Label        24006                          24000
      Group ID     0x800164c                      0x80002f4
      Interface    FortyGigE0/9/0/9.1             FortyGigE0/8/0/9.1
      MTU          9202                           9202
      Control word disabled                       disabled
      PW type      Ethernet                       Ethernet
      VCCV CV type 0x2                            0x2
                   (LSP ping verification)        (LSP ping verification)
      VCCV CC type 0x6                            0x6
                   (router alert label)           (router alert label)
                   (TTL expiry)                   (TTL expiry)
      ------------ ------------------------------ -----------------------------
    Incoming Status (PW Status TLV):
      Status code: 0x0 (Up) in Notification message
    Outgoing Status (PW Status TLV):
      Status code: 0x0 (Up) in Notification message
    MIB cpwVcIndex: 3221225473
    Create time: 21/03/2017 13:09:36 (17:32:46 ago)
    Last time status changed: 22/03/2017 06:29:19 (00:13:03 ago)
    Last time PW went down: 21/03/2017 15:31:24 (15:10:58 ago)
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0
```

Example 2:

Requested bandwidth is not available. In this scenario the PW is down.

```
Group VPWS, XC p1, state is down; Interworking none
  AC: FortyGigE0/9/0/9.1, state is up
    Type VLAN; Num Ranges: 0
    MTU 9202; XC ID 0x256; interworking none
    Statistics:
      packets: received 18016128, sent 97172
      bytes: received 2288436524, sent 444659512
      drops: illegal VLAN 0, illegal length 0
  PW: neighbor 3.3.3.3, PW ID 1, state is down ( all ready )
    PW class vpws1, XC ID 0xc0000001
    Encapsulation MPLS, protocol LDP
    Source address 1.1.1.1
    PW type Ethernet, control word disabled, interworking none
    PW backup disable delay 0 sec
```

```
Sequencing not set
Preferred path tunnel TE 1, fallback enabled
Required BW = 475000 Admited BW = 0
PW Status TLV in use
  MPLS         Local                          Remote
  ------------ ------------------------------ -----------------------------
  Label        24007                          24001
  Group ID     0x8000d7c                      0x80018fc
  Interface    FortyGigE0/9/0/9.1             FortyGigE0/8/0/9.1
  MTU          9202                           9202
  Control word disabled                       disabled
  PW type      Ethernet                       Ethernet
  VCCV CV type 0x2                            0x2
               (LSP ping verification)        (LSP ping verification)
  VCCV CC type 0x6                            0x6
               (router alert label)           (router alert label)
               (TTL expiry)                   (TTL expiry)
  ------------ ------------------------------ -----------------------------
Incoming Status (PW Status TLV):
  Status code: 0x0 (Up) in Notification message
Outgoing Status (PW Status TLV):
  Status code: 0x10 (PW Down) in Notification message
MIB cpwVcIndex: 3221225473
Create time: 25/03/2017 19:09:14 (1d14h ago)
Last time status changed: 27/03/2017 09:23:23 (00:00:03 ago)
Last time PW went down: 27/03/2017 09:23:23 (00:00:03 ago)
Statistics:
  packets: received 97172, sent 18016128
  bytes: received 444659512, sent 2288436524
```

VPWS and PW Scale Details

Scale details for VPWS and PW:

Table 1: Supported LSPs for VPWS and PW

VPWS over physical interface	8000 LSPs
VPWS over bundle interface	1000 LSPs

MPLS Label Distribution Protocol (LDP) Overview

Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) provides the means for peer label switch routers (LSRs) to request, distribute, and release label prefix binding information to peer routers in a network. LDP enables LSRs to discover potential peers and to establish LDP sessions with those peers for the purpose of exchanging label binding information.

MPLS LDP enables one LSR to inform another LSR of the label bindings it has made. Once a pair of routers communicate the LDP parameters, they establish a label-switched path (LSP). MPLS LDP enables LSRs to distribute labels along normally routed paths to support MPLS forwarding. This method of label distribution is also called hop-by-hop forwarding. With IP forwarding, when a packet arrives at a router the router looks at the destination address in the IP header, performs a route lookup, and forwards the packet to the next hop. With MPLS forwarding, when a packet arrives at a router the router looks at the incoming label, looks up the label in a table, and then forwards the packet to the next hop. MPLS LDP is useful for applications that require hop-by-hop forwarding, such as MPLS VPNs.

When you enable MPLS LDP, the LSRs send out messages to try to find other LSRs with which they can create LDP sessions. LDP sessions can be Directly Connected MPLS LDP Sessions or Nondirectly Connected MPLS **LDP Sessions**.

In a **Directly Connected MPLS LDP Session**, LSR is one hop from its neighbor, it is directly connected to its neighbor. The LSR sends out LDP link Hello messages as User Datagram Protocol (UDP) packets to all the routers on the subnet (multicast). A neighboring LSR may respond to the link Hello message, allowing the two routers to establish an LDP session. This is called basic discovery.

In a **Nondirectly Connected MPLS LDP Session**, LSR is more than one hop from its neighbor, it is non-directly connected to its neighbor. For these non-directly connected neighbors, the LSR sends out a targeted Hello message as a UDP packet, but as a unicast message specifically addressed to that LSR. The nondirectly connected LSR responds to the Hello message and the two routers begin to establish an LDP session. This is called extended discovery. **An MPLS LDP targeted session** is a label distribution session between routers that are not directly connected. When you create an MPLS traffic engineering tunnel interface, you need to establish a label distribution session between the tunnel head-end and the tail-end routers. You establish nondirectly connected MPLS LDP sessions by enabling the transmission of targeted Hello messages.

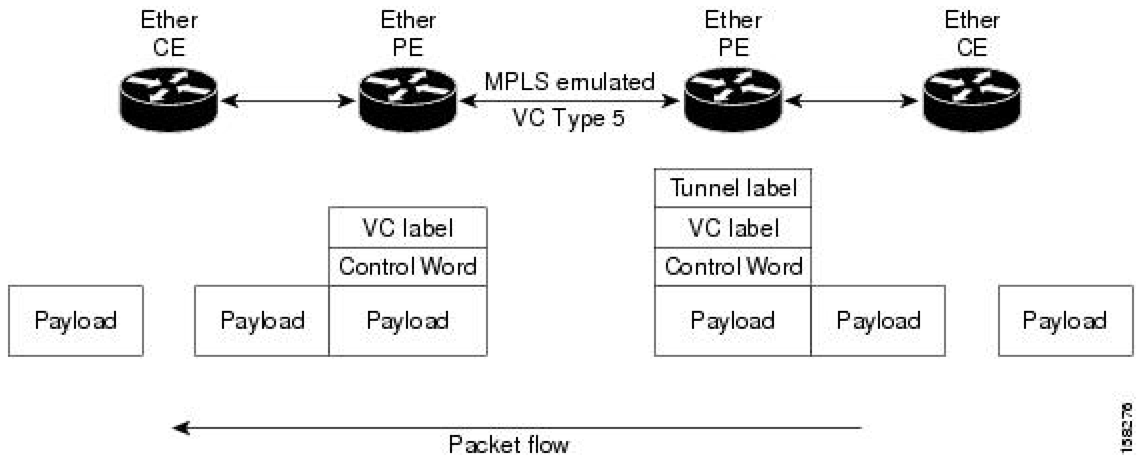
Note Only MPLS LDP targeted sessions are supported.

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.

The following figure provides an example of Ethernet port mode.

Figure 3: Ethernet Port Mode Packet Flow



Type 5 Pseudo Wires

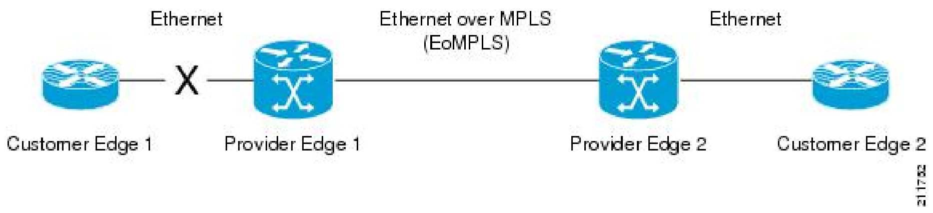
A type 5 PW is known as an Ethernet port-based PW. The ingress PE transports frames received on a main interface or after the subinterface tags have been removed when the packet is received on a subinterface. There is no requirement to send a tagged frame over a type 5 PW, and no dummy tag is added by the EVC-based platforms. The EVC-based platforms have the ability to manipulate the VLAN tags received on the incoming frame with the rewrite command. The results of that VLAN manipulation are transported over the type 5 PW, whether tagged or untagged.

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 NCS 4000 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 failed remote link, especially if the link was 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 4: Remote Link Outage in EoMPLS Wide Area Network



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 interface

To enable this functionality, use the**l2transport propagate** command .

Example

The following example shows how to propagate remote link status changes:

```
RP/0/RP0:hostname# configure
 RP/0/RP0:hostname(config)# interface TenGigE0/3/0/11
 RP/0/RP0:hostname(config-if)# l2transport propagate remote-status
```

VPWS - Sample Configuration Workflow

Complete these configurations on the provider edge routers to enable VPWS.

Topology

```
--(Te0/3/0/11)-[PE1]-(Hu0/14/0/0)-----(Hu0/5/0/0)-[PE2]-(Te0/5/0/9)--
```

where:

- TenGigE0/3/0/11 and TenGigE0/5/0/9 are the access or customer interfaces
- HundredGigE0/14/0/0 and HundredGigE0/5/0/0 are the core interfaces
- PE1 and PE2 are the two L2VPN provider edge (PE) routers. The two PEs are typically connected at two different sites with an MPLS core between them. The attachment circuits (ACs)connected at each L2VPN PE are linked by a pseudowire (PW) over the MPLS network.

Task 1: Bring up the controllers in lan-phy or packet termination mode.

Sample Configuration on PE1	Sample Configuration on PE2

`!`
 	
`controller Optics0/3/0/11`
 	
`port-mode Ethernet framing packet rate 10GE`
 	
`no shut`
 	
`!`
`controller Optics0/14/0/0`
 	
`port-mode Ethernet framing packet rate 100GE`
 	
`no shut`
 	
`!`
	`!`
	
	`controller Optics0/5/0/9`
	
	`port-mode Ethernet framing packet rate 10GE`
	
	`no shut`
	
	`!`
	`controller Optics0/5/0/0`
	
	`port-mode Ethernet framing packet rate 100GE`
	
	`no shut`
	
	`!`

Task 2: Bring up the access and core interfaces.

Sample Configuration on PE1	Sample Configuration on PE2

Access interface:
`interface TenGigE0/3/0/11`
`!`
`interface TenGigE0/3/0/11.1 l2transport`
`encapsulation dot1q 1`
`no shut`
`!`
`interface TenGigE0/3/0/11.2 l2transport`
`encapsulation dot1q 2`
`no shut`
`!`
Core interface:
`interface HundredGigE0/14/0/0`
`ipv4 address 1.76.1.1 255.255.255.0`
` !`
`!`
	Access interface:
	`interface TenGigE0/5/0/9`
	`!`
	`interface TenGigE0/5/0/9.1 l2transport`
	`encapsulation dot1q 1`
	`no shut`
	`!`
	`interface TenGigE0/5/0/9.2 l2transport`
	`encapsulation dot1q 2`
	`no shut`
	`!`
	Core interface:
	`interface HundredGigE0/5/0/0`
	`ipv4 address 1.76.1.2 255.255.255.0`
	` !`
	`!`

Details: Two access interfaces are brought up so that two pseudowires can be created.	

Task 3: Define loopback address.

Sample Configuration on PE1	Sample Configuration on PE2

`!`
`interface Loopback0`
` ipv4 address 1.1.1.1 255.255.255.255`
`!`
	`!`
	`interface Loopback0`
	` ipv4 address 3.3.3.3 255.255.255.255`
	`!`

Task 4: Configure the routing process using OSPF or ISIS on the core interface.

Sample Configuration on PE1	Sample Configuration on PE2

`router ospf 100`
`router-id 1.1.1.1`
`nsf`
`nsr`
` area 0`
` mpls traffic-eng`
` interface Loopback0`
`  !`
`  interface HundredGigE0/14/0/0`
`  !`
` !`
` mpls traffic-eng router-id Loopback0`
` !`
	`router ospf 100`
	`router-id 3.3.3.3`
	`nsf`
	`nsr`
	` area 0`
	` mpls traffic-eng`
	` interface Loopback0`
	`  !`
	`  interface HundredGigE0/5/0/0`
	`  !`
	` !`
	` mpls traffic-eng router-id Loopback0`
	` !`

Details: The sample configuration uses OSPF.	

Task 5: Configure MPLS traffic engineering on the core interface.

Sample Configuration on PE1	Sample Configuration on PE2

`mpls traffic-eng`
`interface HundredGigE0/14/0/0`
` !`
` fault-oam`
` !`
	`mpls traffic-eng`
	`interface HundredGigE0/5/0/0`
	` !`
	` fault-oam`
	` !`

Task 6: Configure RSVP on the core interface.

Sample Configuration on PE1	Sample Configuration on PE2

`rsvp`
`interface HundredGigE0/14/0/0`
`bandwidth 100`
` !`
`!`
	`rsvp`
	`interface HundredGigE0/5/0/0`
	`bandwidth 100`
	` !`
	`!`

Task 7: Configure MPLS OAM for MPLS pseudowires to work on the core interfaces.

Sample Configuration on PE1	Sample Configuration on PE2

`!`
`mpls oam`
`!`
	`!`
	`mpls oam`
	`!`

Task 8: Configure the tunnel interface. It can be a MPLS-TE or Flex-LSP tunnel.

Sample Configuration on PE1	Sample Configuration on PE2

Sample Configuration on PE1	Sample Configuration on PE2
MPLS TE tunnel:	MPLS TE tunnel:
`interface tunnel-te1`	`interface tunnel-te1`
`ipv4 unnumbered Loopback0`	`ipv4 unnumbered Loopback0`
`signalled-bandwidth 1`	`signalled-bandwidth 1`
`destination 3.3.3.3`	`destination 1.1.1.1`
`path-selection`	`path-selection`
`metric te`	`metric te`
`bandwidth 50000`	`bandwidth 50000`
`!`	`!`
` path-option 1 dynamic`	` path-option 1 dynamic`
`!`	`!`
Flex-LSP tunnel with BFD:	Flex-LSP tunnel with BFD:
`interface tunnel-te2`	`interface tunnel-te2`
`ipv4 unnumbered Loopback0`	`ipv4 unnumbered Loopback0`
`bfd`	`bfd`
` encap-mode gal`	` encap-mode gal`
` multiplier 3`	` multiplier 3`
` fast-detect`	` fast-detect`
` minimum-interval 100`	` minimum-interval 100`
` !`	` !`
` signalled-bandwidth 1`	` signalled-bandwidth 1`
` destination 3.3.3.3`	` destination 1.1.1.1`
` bidirectional`	` bidirectional`
` association id 86 source-address 192.0.0.0`	` association id 86 source-address 192.0.0.0`
` association type co-routed`	` association type co-routed`
`   fault-oam`	`   fault-oam`
`  !`	`  !`
` !`	` !`
`path-selection`	`path-selection`
`  metric te`	`  metric te`

Sample Configuration on PE1	Sample Configuration on PE2

`bandwidth 50000`
` !`
` path-option 1 dynamic`
` !`
	`bandwidth 50000`
	` !`
	` path-option 1 dynamic`
	` !`

Details: This is the tunnel bandwidth configuration that is required for VPWS CAC to work. The pseudowire requested bandwidth must be within the tunnel bandwidth value.	

Task 9: Setup interfaces running LDP:

Sample Configuration on PE1	Sample Configuration on PE2

`mpls ldp`
`nsr`
`log`
`  neighbor`
`   nsr`
`  graceful-restart`
`  !`
`graceful-restart reconnect-timeout 169`
`graceful-restart forwarding-state-holdtime 180`
`discovery`
`targeted-hello holdtime 180`
`targeted-hello interval 20`
`!`
`router-id 1.1.1.1`
`session protection`
`address-family ipv4`
`discovery targeted-hello accept`
` !`
`!`
	`mpls ldp`
	`nsr`
	`log`
	`  neighbor`
	`   nsr`
	`  graceful-restart`
	`  !`
	`graceful-restart reconnect-timeout 169`
	`graceful-restart forwarding-state-holdtime 180`
	`discovery`
	`targeted-hello holdtime 180`
	`targeted-hello interval 20`
	`!`
	`router-id 3.3.3.3`
	`session protection`
	`address-family ipv4`
	`discovery targeted-hello accept`
	` !`
	`!`

Details: The two PEs establish a targeted MPLS LDP session between themselves so they can establish and control the status of the pseudowire. The targeted MPLS LDP session is established over MPLS-TE or Flex LSP.	

Task 10: Configure VPWS static and dynamic pseudowires.

Sample Configuration on PE1	**Sample Configuration on PE2**

Pseudowire 1 (vpws-pw-1) uses MPLS-TE tunnel (tunnel-te 1):
`l2vpn`
`pw-class vpws-pw-1`
`encapsulation mpls`
`protocol ldp`
`ipv4 source 1.1.1.1`
`preferred-path interface tunnel-te 1`
` !`
	Pseudowire 1 (vpws-pw-1) uses MPLS-TE tunnel (tunnel-te 1):
	`l2vpn`
	`pw-class vpws-pw-1`
	`encapsulation mpls`
	`protocol ldp`
	`ipv4 source 3.3.3.3`
	`preferred-path interface tunnel-te 1`
	` !`

Pseudowire 2 (vpws-pw-2) uses Flex-LSP tunnel (tunnel-te 2):
`!`
` pw-class vpws-pw-2`
` encapsulation mpls`
` protocol ldp`
` ipv4 source 1.1.1.1`
` preferred-path interface tunnel-te 2`
`!`
	Pseudowire 2 (vpws-pw-2) uses Flex-LSP tunnel (tunnel-te 2):
	`!`
	` pw-class vpws-pw-2`
	` encapsulation mpls`
	` protocol ldp`
	` ipv4 source 3.3.3.3`
	` preferred-path interface tunnel-te 2`
	`!`

Configure pseudowire 1 (vpws-pw-1) as dynamic:
`!`
` xconnect group vpws`
` p2p pw1`
` interface TenGigE0/3/0/11.1`
` neighbor ipv4 3.3.3.3 pw-id 1`
` bandwidth 1000`
` pw-class vpws-pw-1`
`  !`
	Configure pseudowire 1 (vpws-pw-1) as dynamic:
	`!`
	` xconnect group vpws`
	` p2p pw1`
	` interface TenGigE0/5/0/9.1`
	` neighbor ipv4 1.1.1.1 pw-id 1`
	` bandwidth 1000`
	` pw-class vpws-pw-1`
	`  !`

Sample Configuration on PE1	**Sample Configuration on PE2**

Configure pseudowire 2(vpws-pw-2) as static:
 	
`!`
`p2p pw2`
`interface TenGigE0/3/0/11.2`
`neighbor ipv4 3.3.3.3 pw-id 2`
`mpls static label local 100 remote 200`
`bandwidth 1000`
`pw-class vpws-pw-2`
`   !`
`!`
	Configure pseudowire 2(vpws-pw-2) as static:
	
	`!`
	`p2p pw2`
	`interface TenGigE0/5/0/9.2`
	`neighbor ipv4 1.1.1.1 pw-id 2`
	`mpls static label local 100 remote 200`
	`bandwidth 1000`
	`pw-class vpws-pw-2`
	`   !`
	`!`

Details: The bandwidth command is used to allocate bandwidth to the pseudowire for VPWS CAC to function .	

High Availability

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

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 targeted 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). Preferred tunnel path configuration applies only to MPLS encapsulation.

Understanding 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.

Configuring L2VPN Interface or Connection for L2VPN

Perform this task to configure an interface or a connection for L2VPN.

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0:hostname# configure
```

Enters global configuration mode.

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

Example:

```
RP/0/RP0:hostname(config)# interface TenGigE0/6/0/6.11 l2transport
```

Enters interface configuration mode, configures an interface and enables L2 transport on the interface.

Step 3 **exit**

Example:

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

Exits the configuration mode.

Step 4 **interface** *type interface-path-id*

Example:

```
RP/0/RP0:hostname(config)# interface TenGigE0/6/0/6.11
```

Enters interface configuration mode and configures an interface.

Step 5 **end or commit**

Example:

```
RP/0/RP0:hostname(config-if)# end
or
RP/0/RP0:hostname(config-if)# commit
```

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

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 6 **show interface** *type interface-id*

Example:

```
RP/0/RP0:hostname# show interface TenGigE0/6/0/6.11
```

(Optional) Displays the configuration settings you committed for the interface.

Configuring Local Switching

Perform this task to configure local switching.

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0:hostname# configure
```

Enters global configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0:hostname# l2vpn
```

Enters L2VPN configuration mode.

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

Example:

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

Enters the name of the cross-connect group.

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

Example:

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

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

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

Example:

```
RP/0/RP0:hostname(config-l2vpn-xc-p2p)# interface TenGigE0/6/0/2.10
```

Specifies the interface type ID. The choices are:

- TenGigE: TenGigabit Ethernet/IEEE 802.3 interfaces.
- HundredGigE: Hundred Gigabit Ethernet/IEEE 802.3 interfaces.
- CEM: Circuit Emulation interface.

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

Example:

```
RP/0/RP0:hostname(config-l2vpn-xc-p2p)# interface HundredGigE0/3/0/0.30
```

Specifies the interface type ID. The choices are:

- TenGigE: TenGigabit Ethernet/IEEE 802.3 interfaces.
- HundredGigE: Hundred Gigabit Ethernet/IEEE 802.3 interfaces.

Step 7 **end or commit**

Example:

```
RP/0/RP0:hostname(config-if)# end
or
RP/0/RSP0/CPU0:router(config-if)# commit
```

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

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Static Point to Point Cross-Connect

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

Please 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.

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0:hostname# configure
```

Enters global configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0:hostname(config)# l2vpn
```

Enters L2VPN configuration mode.

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

Example:

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

Enters the name of the cross-connect group.

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

Example:

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

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

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

Example:

```
RP/0/RP0:hostname(config-l2vpn-xc-p2p)# interface TenGigE0/6/0/2.10
```

Specifies the interface type and instance.

Step 6 **neighbor** A.B.C.D **pw-id** pseudowire-id

Example:

```
RP/0/RP0:hostname(config-l2vpn-xc-p2p)# neighbor 10.2.2.2 pw-id 2000
```

Configures the pseudowire segment for the cross-connect.

Use the A.B.C.D argument to specify the IP address of the cross-connect peer.

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

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:hostname(config-l2vpn-xc-p2p-pw)# mpls static label local 699 remote 890
```

Configures local and remote label ID values.

Step 8 **end or commit**

Example:

```
RP/0/RP0:hostname(config-if)# end
or
RP/0/RP0:hostname(config-if)# commit
```

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

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Dynamic Point to Point Cross-Connect

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

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

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0:hostname# configure
```

Enters global configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0:hostname(config)# l2vpn
```

Enters L2VPN configuration mode.

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

Example:

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

Enters the name of the cross-connect group.

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

Example:

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

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

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

Example:

```
RP/0/RP0:hostname(config-l2vpn-xc-p2p)# interface TenGigE0/6/0/2.10
```

Specifies the interface type ID. The choices are:

- TenGigE: TenGigabit Ethernet/IEEE 802.3 interfaces.
- HundredGigE: Hundred Gigabit Ethernet/IEEE 802.3 interfaces.
- CEM: Circuit Emulation interface.

Step 6 **neighbor** A.B.C.D **pw-id** pseudowire-id

Example:

```
RP/0/RP0:hostname(config-l2vpn-xc-p2p)# neighbor 10.2.2.2 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 **end or commit**

Example:

```
RP/0/RP0:hostname(config-if)# end
or
RP/0/RP0:hostname(config-if)# commit
```

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

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring L2VPN Quality of Service

This section describes how to configure L2VPN quality of service (QoS) in port mode.

Configuring L2VPN Quality of Service Policy in Port Mode

This section describes how to configure L2VPN quality of service (QoS) in port mode.

Note In port mode, the interface name format does not include a subinterface number; for example, TenGigE0/3/0/5.20.

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0:hostname# configure
```

Enters global configuration mode.

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

Example:

```
RP/0/RP0:hostname(config)# interface TenGigE0/3/0/5.20
```

Specifies the interface attachment circuit.

Step 3 **l2transport**

Example:

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

Configures an interface or connection for L2 switching.

Step 4 **service-policy** [**input** | **output**] [policy-map-name]

Example:

```
RP/0/RP0:hostname(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 5 **end or commit**

Example:

```
RP/0/RP0:hostname(config-if)# end
or
RP/0/RP0:hostname(config-if)# commit
```

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

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Preferred Tunnel Path

This procedure describes how to configure a preferred tunnel path.

Note The tunnel used for the preferred path configuration is an MPLS Traffic Engineering (MPLS-TE) tunnel.

Procedure

Step 1 **configure**

Example:

```
RP/0/RP0:hostname# configure
```

Enters global configuration mode.

Step 2 **l2vpn**

Example:

```
RP/0/RP0:hostname(config)# l2vpn
```

Enters L2VPN configuration mode.

Step 3 **pw-class** *name*

Example:

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

Configures the pseudowire class name.

Step 4 **encapsulation mpls**

Example:

```
RP/0/RP0:hostname(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:hostname(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/TP tunnel is configured as the preferred path goes down, the corresponding pseudowire can also go down.

Note Ensure that fallback is supported.

Step 6 **end or commit**

Example:

```
RP/0/RP0:hostname(config-if)# end
or
RP/0/RP0:hostname(config-if)# commit
```

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

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:

  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.