EVPN MPLS Single-Homing

This chapter provides comprehensive guidance on configuring and deploying EVPN MPLS single-homing, including both E-LAN and E-Line services. Users can learn about supported features, benefits, key workflows, and step-by-step configuration procedures for single-homed connectivity, traffic engineering, and private line emulation in Cisco MPLS networks.

EVPN MPLS single-homing mode

EVPN MPLS single-homing is a network connectivity mode that

- enables a single connection between a customer edge (CE) device and a provider edge (PE) device using MPLS as the transport protocol
- simplifies network design by eliminating the need for redundant connections at the CE level, and
- provides efficient Layer 2 and Layer 3 connectivity over an MPLS backbone.

EVPN MPLS single-homing is a cost-effective and simplified network solution designed for scenarios prioritizing cost efficiency and simplicity over redundancy. This mode is commonly used in environments where the customer edge (CE) device does not require a backup connection to the provider edge (PE) device, and is ideal for non-critical applications, such as small enterprises connecting branch offices to an MPLS network, as it reduces costs, simplifies deployment, and minimizes operational complexity.

EVPN MPLS single-homing mode services

The EVPN MPLS single-homing mode supports various service types to provide diverse network connectivity options. These services are designed to replace legacy technologies, offering enhanced flexibility, scalability, and fault tolerance. Both services utilize the EVPN control plane for scalability, redundancy, and efficient MAC address learning.

The EVPN MPLS single-homing mode supports these services:

- EVPN E-LAN—EVPN E-LAN offers multipoint-to-multipoint Layer 2 connectivity, enabling seamless communication among multiple sites as if they were on the same LAN. This solution is particularly suitable for scenarios requiring interconnectivity among more than two locations, such as data center interconnects or corporate WANs. For example, a company with offices in New York, London, and

Tokyo can utilize EVPN E-LAN to interconnect these offices, allowing all sites to communicate as part of the same Ethernet network.

- EVPN E-Line—EVPN E-Line provides point-to-point Layer 2 connectivity between two endpoints, such as customer sites or data centers, making it an ideal solution for scenarios where direct communication is required over a service provider's MPLS or IP backbone. For instance, a business with offices in two cities can leverage EVPN E-Line to establish a direct, transparent Ethernet connection between the sites, ensuring seamless and efficient communication.

This table compares the key features and use cases of EVPN E-LAN and EVPN E-Line services.

Table 1: EVPN E-LAN and EVPN E-Line services

Feature	EVPN E-LAN	EVPN E-Line
Connectivity type	Multipoint-to-Multipoint	Point-to-Point
Typical use cases	Multi-site connectivity, interconnecting multiple locations	Data center interconnects, connecting two sites
MAC address learning	Across all endpoints in the E-LAN	Limited to two endpoints

EVPN E-LAN single-homing

A EVPN E-LAN single-homing is a network connectivity model that

- simplifies network infrastructure by connecting a Layer 2 gateway device to a single access network
- reduces costs by eliminating the need for additional infrastructure, and
- supports seamless communication between customer sites over an MPLS network.

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
EVPN E-LAN Single-Homing	Release 24.4.1	Introduced in this release on: Fixed Systems (8700) (select variants only*) * The EVPN E-LAN single-homing functionality is now extended to the Cisco 8712-MOD-M routers.

Feature Name	Release Information	Feature Description
EVPN E-LAN Single-Homing	Release 24.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) * The EVPN E-LAN single-homing functionality is now extended to these fixed systems and line cards. • 8212-48FH-M • 8711-32FH-M • 88-LC1-52Y8H-EM • 88-LC1-12TH24FH-E
EVPN E-LAN Single-Homing	Release 24.2.11	Introduced in this release on: Modular Systems (8800 [LC ASIC: P100]) (select variants only*) * The EVPN E-LAN single-homing functionality is now extended to the Cisco 88-LC1-36EH routers.
EVPN E-LAN Single-Homing	Release 7.11.1	We now offer a cost-effective and simplified solution for seamless communication between various customer sites connected to the same service provider network using Ethernet Virtual Private Network (EVPN) single-homing mode. EVPN LAN (E-LAN) is a service to bridge Ethernet data traffic among different sites across the MPLS network connecting a Layer 2 gateway device to a single access network. In the single-homing mode, a device is connected to one router in the MPLS core through physical ports or bundle ports, and in the event of a failure on those links, the traffic over the links is not protected by links to another router on the core. This feature is supported only on Q200-based line cards. The feature introduces the **evpn** commands.

The EVPN E-LAN single-homing delivers seamless multipoint-to-multipoint Layer 2 connectivity, enabling efficient communication between multiple endpoints such as customer sites or data centers. Designed for scenarios requiring transparent Ethernet connectivity over a service provider's MPLS or IP backbone, this solution ensures reliable and scalable data exchange, making it an ideal choice for businesses aiming to interconnect multiple locations with ease.

Scenarios for deploying EVPN E-LAN single-homing

These scenarios outline the ideal use cases for deploying EVPN E-LAN single-homing:

- Small branch offices—cost-effective solution for small remote sites or branch offices with minimal traffic or redundancy requirements.
- Initial deployments—suitable for organizations starting their EVPN journey, with the flexibility to scale to multi-homing later.

Benefits of EVPN E-LAN single-homing

The EVPN E-LAN single-homing offers these benefits:

- Simplified design—reduces network complexity by using a single CE-to-PE connection, eliminating the need for redundancy protocols like MC-LAG.
- Cost-effectiveness—minimizes physical links and hardware requirements, making it ideal for small sites or budget-conscious scenarios.
- Efficient resource utilization—consumes less bandwidth and hardware compared to multi-homing, making it suitable for low-traffic or non-critical environments.
- Ease of management—simplifies troubleshooting and operations with single PE-CE connections, removing the need for configuring load balancing or redundancy mechanisms.
- Multipoint connectivity— enables full multipoint-to-multipoint connectivity across all sites, making it perfect for collaborative workspaces or shared data centers.
- Interoperability—ensures compatibility across multi-vendor environments through open standards, providing a flexible foundation for scaling to multi-homing in the future.
- Centralized control—utilizes BGP signaling for efficient MAC address distribution and control, while distributed forwarding ensures effective data handling.
- Scalability—supports large-scale deployments for small or remote sites.

How EVPN E-LAN single-homing transports traffic

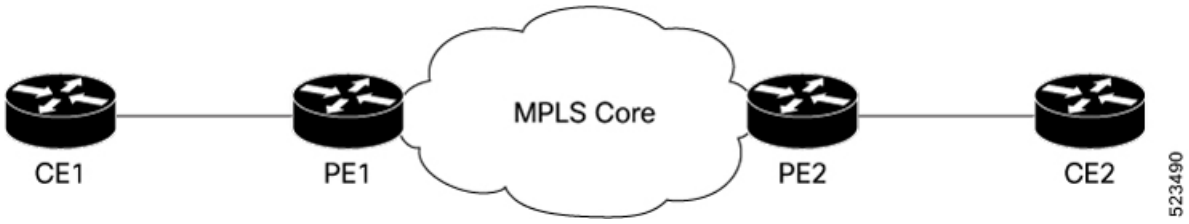
Summary

The EVPN E-LAN single-homing process facilitates the transport of Layer 2 traffic between customer sites by utilizing EVPN control plane protocols. This process ensures efficient traffic forwarding and seamless MAC address distribution across the network.

The key components involved in the process are:

- CE1: Sends Layer 2 traffic to PE1.
- PE1: Receives and processes Layer 2 frames from CE1, determines the destination, and distributes MAC address information.
- PE2: Forwards the Layer 2 frames to the appropriate attachment circuit connected to CE2.
- CE2: Receives the Layer 2 traffic from PE2.

Workflow



The process involves these stages:

1. Traffic transmission from CE1 to PE1
   - CE1 sends Layer 2 traffic to PE1 over a single bundle or physical link.
   - The traffic is encapsulated in Layer 2 frames.
2. Frame processing at PE1
   - PE1 receives the Layer 2 frames on its ingress interface.
   - PE1 checks the destination MAC address of the frame and determines the appropriate attachment circuit for forwarding.
3. MAC address distribution through EVPN control plane
   - PE1 uses EVPN control plane protocols to distribute the MAC address information learned from CE1 to PE2.
4. Traffic forwarding by PE2
   - PE2, having obtained the destination MAC address from the EVPN control plane, forwards the Layer 2 frames to the appropriate attachment circuit connected to CE2.

The process ensures seamless and efficient transport of Layer 2 traffic between customer sites, leveraging EVPN protocols for MAC address distribution and traffic.

Configure EVPN E-LAN single-homing

Enable EVPN E-LAN single-homing on PE1 to advertise MAC addresses and distribute MAC address information from CE1 to PE2.

This task involves configuring Ethernet VPN Identifier (EVI) under the bridge domain and ensuring PE1 advertises MAC addresses. The configuration disables EVPN multi-homing on the bundle interface and sets up BGP for L2VPN and EVPN.

Follow these steps to configure EVPN E-LAN single-homing:

Before you begin

- Ensure PE1 is connected to CE1 and PE2.
- Verify that the required software version supports EVPN E-LAN single-homing.

Procedure

Step 1 Disable EVPN multi-homing.

By default, the bundle interface is in EVPN multi-homing mode. To disable EVPN multi-homing, configure bundle-Ether AC with ESI value (identifier type) set to zero.

Example:

```
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether1
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.00.00.00.00.00.00.00.00
```

Alternatively, disable EVPN multi-homing globally by:

```
Router(config)# evpn
Router(config-evpn)# ethernet-segment type 1 auto-generation-disable
```

Step 2 Set up BGP for L2VPN and EVPN.

```
Router# configure
Router#(config)# router bgp 200
Router#(config-bgp)# bgp router-id 10.10.10.1
Router#(config-bgp)# address-family l2vpn evpn
Router#(config-bgp)# neighbor 10.10.10.10
Router#(config-bgp-nbr)# remote-as 200
Router#(config-bgp-nbr)# update-source Loopback 0
Router#(config-bgp-nbr)# address-family l2vpn evpn
```

Step 3 Configure the bridge domain.

```
Router(config)# l2vpn
Router (config-l2vpn)# bridge group BG1
Router (config-l2vpn-bg)# bridge-domain BD1
Router (config-l2vpn-bg-bd)# interface Bundle-Ether1.2001
Router (config-l2vpn-bg-bd-ac)# evi 2001
```

Step 4 Configure MAC advertisement.

```
Router(config)# evpn
Router(config-evpn)# evi 2001
Router(config-evpn-instance)# advertise-mac
Router(config-evpn-instance-mac)# commit
```

Step 5 Running configuration of EVPN E-LAN single-homing mode.

```
router bgp 200
 bgp router-id 10.10.10.1
 address-family l2vpn evpn
 neighbor 10.10.10.10
  remote-as 200 description MPLS-FACING-PEER
  update-source Loopback0
  address-family l2vpn evpn
 !

l2vpn
 bridge group BG1
  bridge-domain BD1
   interface BundleEther1.2001
    evi 2001
   !

evpn
 interface Bundle-Ether 1
    ethernet-segment
    identifier type 0 00.00.00.00.00.00.00.00.00
   !
evi 2001
 advertise-mac
!
```

Step 6 Use the **show evpn ethernet-segment interface Bundle-Ether 1 detail** command to verify that EVPN E-LAN single-homing mode is configured.

Example:

In this example, the operational mode is SH or single-homing, which indicates that CE1 is connected to PE1 through a single link.

```
Router# show evpn ethernet-segment interface Bundle-Ether 1 detail

Ethernet Segment Id    Interface        Nexthops
---------------------  ----------       ----------
N/A                    Bundle-Ether 1   10.0.0.2
……
Topology :
Operational : SH
```

After the configuration is complete, PE1 operates in single-homing mode (SH) and advertise MAC addresses learned from CE1 to PE2.

EVPN E-Line single-homing

An EVPN E-Line single-homing is a network connectivity mode that

- provides point-to-point Ethernet connectivity between two customer endpoints
- operates over a single physical connection to the PE router, and
- ensures simplicity and cost-effectiveness for scenarios where redundancy is not required.

EVPN E-Line operates over both IP and MPLS cores: the IP core supports BGP functionality, while the MPLS core enables efficient packet switching between endpoints.

Table 3: Feature History Table

Feature Name	Release Information	Feature Description
EVPN E-Line Single-Homing	Release 24.4.1	Introduced in this release on: Fixed Systems (8700) (select variants only*) * The EVPN E-Line single-homing is now extended to the Cisco 8712-MOD-M routers.

EVPN E-Line Single-Homing	Release 24.3.1	Introduced in this release on: Fixed Systems (8200, 8700); Modular Systems (8800 [LC ASIC: P100]) (select variants only*) * The EVPN E-Line single-homing is now extended to: • 8212-48FH-M • 8711-32FH-M • 88-LC1-52Y8H-EM • 88-LC1-12TH24FH-E
EVPN E-Line Single-Homing	Release 24.2.11	Introduced in this release on: Modular Systems (8800 [LC ASIC: Q200, P100]) (select variants only*) * The EVPN E-Line single-homing is now extended to routers with the Q200 and 88-LC1-36EH line cards.
EVPN E-Line Single-Homing	Release 7.8.1	The EVPN E-Line single-homing is a BGP control plane solution for point-to-point services. It implements the signaling and encapsulation techniques for establishing an EVPN instance between a pair of PEs. It provides the service of forwarding L2 Ethernet traffic between network devices without inspecting the MAC header in the Ethernet frame. The use of EVPN for VPWS eliminates the need for signaling single-segment and multi-segment pseudowire (PW) for point-to-point Ethernet services.

The EVPN E-Line single-homing is a network service mode that provides point-to-point Ethernet connectivity between two customer endpoints using a single physical connection to the PE router. It is designed for simplicity and cost-effectiveness, making it ideal for small-scale deployments or non-critical applications where redundancy is not required. This service leverages EVPN technology to deliver reliable Layer 2 VPN services over an IP-MPLS network, adhering to the E-Line service type defined by the Metro Ethernet Forum (MEF).

Benefits EVPN E-Line single-homing

The EVPN E-Line single-homing mode offers these benefits:

- Scalability—achieves scalability without the need for signaling pseudowires, simplifying network operations.
- Ease of provisioning—simplifies the setup process, reducing the time and effort required for deployment.
- Elimination of pseudowires—operates without pseudowires (PWs), streamlining the network architecture.
- Optimal forwarding—leverages BGP's best path selection mechanism to ensure efficient and optimal data forwarding.

Prerequisites for E-Line single-homing

To configure EVPN E-Line, ensure these prerequisites are met:

- BGP configuration: Verify that BGP is configured to support EVPN Subsequent Address Family Identifier (SAFI).
- BGP session: Establish a BGP session between PEs with the `address-family l2vpn evpn` configuration to enable the exchange of EVPN routes.

Restrictions for EVPN E-Line single-homing

These restrictions ensure compatibility, interoperability, and adherence to system requirements and are critical for ensuring proper configuration and avoiding operational issues in EVPN E-Line single-homing deployments.

- VPN ID uniqueness—The VPN ID must be unique per router.
- Route target uniqueness—When specifying a list of route targets, they must be unique per PE (per BGP address-family).
- MTU signaling and enforcement:
  - On versions earlier than IOS XR Release 7.0.x—MTU is not signaled, and MTU mismatches are ignored without interoperability issues.
  - On versions later than IOS XR Release 7.0.x—L3 MTU is advertised by default, and MTU mismatches are enforced by default. However, interoperability issues arise with IOS XR Release 7.3.2 if **transmit-l2-mtu** is configured, as L3 and L2 MTUs do not match. To avoid this, configure the **transmit-mtu-zero** and **ignore-mtu-mismatch** commands.
  - On versions later than IOS XR Release 7.3.2—MTU of 0 is advertised by default, and MTU mismatches are ignored by default. L2 MTU can be advertised using the **transmit-l2-mtu** command, and MTU mismatches can be enforced with the **enforce-mtu-mismatch** command.
- Pseudowire Headend (PWHE) configuration—EVPN E-Line does not support PWHE configuration.
- On versions prior to IOS XR Release 24.1.1:
  - EVPN E-line is supported only on single-homing and is not supported on dual homing. This is applicable for both the local and remote sides of the network.
  - EVPN validates if the route is for a single home next hop, otherwise it issues an error message. An Ethernet Segment Identifier (ESI) is an attribute that is used to enable EVPN multi-homing. EVPN relies on the ESI value being zero to determine if this is a single home or not. If the AC is a Bundle-Ether interface running LACP, then you need to manually configure the ESI value to zero to overwrite the auto-sense ESI, as EVPN-VPWS multi-homing is not supported.

    To disable EVPN dual homing, configure bundle-Ether AC with ESI value (**identifier type**) set to zero.

    ```
    Router(config)# evpn
    Router(config-evpn)# interface Bundle-Ether12
    Router(config-evpn-ac)# ethernet-segment
    Router(config-evpn-ac-es)# identifier type 0 00.00.00.00.00.00.00.00.00
    ```

    As an alternative, you can disable EVPN dual homing globally.

    ```
    Router(config)# evpn
    Router(config-evpn)# ethernet-segment type 1 auto-generation-disable
    ```

How EVPN-Line single-homing works

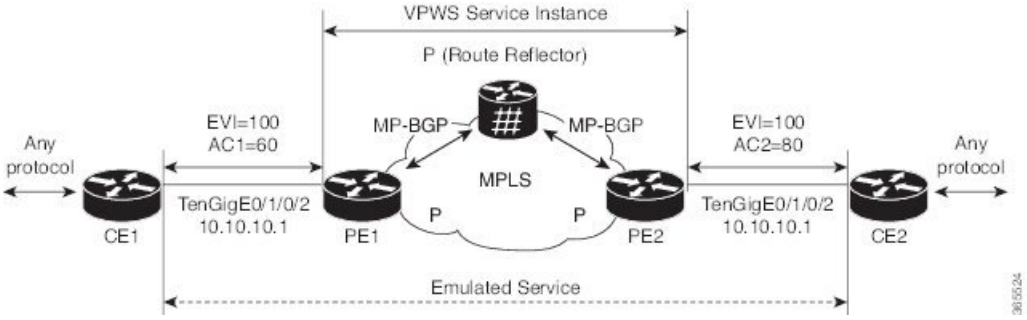
Summary

The EVPN-Line single-homing process enables EVPN services for point-to-point connections using a single PE device. This process ensures efficient communication between local and remote endpoints by leveraging multi-protocol BGP and MPLS labels.

The key components involved in the process are:

- PE devices: PE devices that run multi-protocol BGP to manage EVPN routes and connect to CE routers.
- Ethernet Virtual Instance (EVI): Represents the VPN ID for the service.
- Attachment Circuits (ACs): Local (AC1) and remote (AC2) identifiers for the emulated service endpoints.
- MPLS labels: Allocated per local AC for reachability.
- CE devices: Customer Edge routers (CE1 and CE2) that connect to the service provider network.
- Route Reflector (P): Facilitates MP-BGP signaling within the MPLS network.
- MPLS Network: Provides the transport for the emulated service.

Workflow



The EVPN E-line single-homing process involves these stages:

1. Configuration of PE1 and PE2.
   - PE1:
     - Specify the VPN ID (EVI).
     - Define local and remote AC identifiers (AC1 and AC2).
     - Allocate an MPLS label for the local AC.
   - PE2:
     - Mirror the configuration of PE1 to ensure symmetry.
     - Allocate an MPLS label for the local AC.
2. Route advertisement.

- PE1 advertises a single EVPN per EVI Ethernet Auto Discovery (AD) route for each local endpoint (AC) to remote PEs, including the associated MPLS label.
- PE2 performs the same task.

3. Route reception and database update
   - Upon receiving the EVPN per EVI AD route from PE2, PE1 updates its local EVPN database with the path list to reach AC2 (for example., the next hop is PE2's IP address and MPLS label for AC2).
   - PE2 performs the same task for routes received from PE1.
4. Traffic encapsulation and forwarding
   - CE1 and CE2 communicate using any protocol to send traffic to PE1 and PE2, respectively.
   - PE1 and PE2 encapsulate the traffic and forward it across the MPLS network.
   - MP-BGP is used between PE1, PE2, and the RR (P) to exchange VPWS service instance information.
   - The RR (P) reflects the MP-BGP updates between PE1 and PE2.
   - PE1 and PE2 decapsulate the traffic and forward it to CE1 and CE2, respectively.

   The EVPN E-line single-homing process enables efficient and reliable communication between customer edge devices over an MPLS network, ensuring seamless data transfer and network stability.

Configure EVPN E-Line single-homing

Configure EVPN E-line single-homing to establish connectivity between PE devices.

Before you begin

- Ensure that the PE devices are connected and reachable.
- Verify that the required interfaces are operational.
- Confirm that the BGP configuration is enabled on both PE devices.

Procedure

Step 1 Configure PE1.

Example:

```
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 10.10.10.1
Router(config-bgp-nbr))# address-family l2vpn evpn
Router(config-bgp-nbr-af)# commit
Router(config-bgp-nbr-af)# root
Router(config)# l2vpn
```

```
Router(config-l2vpn)# xconnect group evpn-vpws
Router(config-l2vpn-xc)# p2p evpn1
Router(config-l2vpn-xc-p2p)# interface TenGigE0/1/0/2
Router(config-l2vpn-xc-p2p)# neighbor evpn evi 100 target 12 source 10
Router(config-l2vpn-xc-p2p-pw)# exit
Router(config-l2vpn-xc-p2p)# commit
```

Step 2 Configure PE2.

Example:

```
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 10.10.10.1
Router(config-bgp-nbr))# address-family l2vpn evpn
Router(config-bgp-nbr-af)# commit
Router(config-bgp-nbr-af)# root
Router(config)# l2vpn
Router(config-l2vpn)# xconnect group evpn-vpws
Router(config-l2vpn-xc)# p2p evpn1
Router(config-l2vpn-xc-p2p)# interface TenGigE0/1/0/2
Router(config-l2vpn-xc-p2p)# neighbor evpn evi 100 target 10 source 12
Router(config-l2vpn-xc-p2p-pw)# exit
Router(config-l2vpn-xc-p2p)# commit
```

If the source and target AC IDs are the same, use the **neighbor evpn evi 100 service 10** command to configure the neighbor EVPN.

Step 3 Running configuration of EVPN E-Line single-homing mode.

Example:

```
/* On PE1 */
configure
router bgp 100
 address-family l2vpn evpn
 neighbor 10.10.10.1
  address-family l2vpn evpn
 !

configure
l2vpn
 xconnect group evpn-vpws
  p2p evpn1
   interface TenGigE0/1/0/2
   neighbor evpn evi 100 target 12 source 10
  !


/* On PE2 */
configure
router bgp 100
 address-family l2vpn evpn
 neighbor 10.10.10.1
  address-family l2vpn evpn
 !

configure
l2vpn
 xconnect group evpn-vpws
  p2p evpn1
```

```
 interface TenGigE0/1/0/2
 neighbor evpn evi 100 target 10 source 12
!
```

Supported EVPN E-Line single-homing features

EVPN E-Line single-homing supports these features:

- Flow label support for EVPN E-Line
- Private line emulation over EVPN E-Line Single-Homing

Flow label support for EVPN E-Line

A flow label support for EVPN E-Line is a feature that

- enables provider (P) routers to use flow-based load balancing to forward traffic between PE devices
- utilizes Flow-Aware Transport (FAT) of pseudowires (PW) over an MPLS packet-switched network for load balancing traffic across BGP-based signaled pseudowires for EVPN E-Line, and
- improves traffic distribution across equal cost multipaths (ECMP) or link-bundled paths in the core.

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
Flow label support for EVPN E-Line	Release 24.4.1	Introduced in this release on: Fixed Systems (8700) (select variants only*) You can now improve traffic distribution by enabling flow-based load balancing between provider edge devices. This feature leverages Flow-Aware Transport (FAT) pseudowires over an MPLS network to efficiently manage traffic across BGP-signaled pseudowires in an Ethernet VPN. By using flow labels derived from packet flows, the router ensures optimal traffic distribution across equal-cost multipaths or link-bundled paths. * This feature is now supported on the Cisco 8712-MOD-M routers.
Flow label support for EVPN E-Line	Release 25.3.1	The Flow Label support improves traffic distribution by enabling flow-based load balancing between provider edge devices. It uses Flow-Aware Transport (FAT) pseudowires over an MPLS network to efficiently manage traffic across BGP-signaled pseudowires in an Ethernet VPN. Flow labels, based on packet flows, help optimize traffic across equal cost multipaths or link-bundled paths.

Optimize traffic distribution using flow label

A service provider deploying EVPN E-Line can use this feature to optimize traffic distribution across multiple paths in their MPLS core network. A network with high traffic volumes can benefit from reduced congestion and improved performance by leveraging flow-based load balancing.

Flow-based load balancing with FAT pseudowires

FAT pseudowires (PWs) provide the capability to identify individual flows within a PW, allowing routers to load-balance traffic effectively. When ECMP is used, FAT PWs ensure better traffic distribution by creating a flow label based on indivisible packet flows entering an imposition PE. This flow label is inserted as the lowermost label in the packet. P routers then use the flow label for load balancing.

A flow is identified by either:

- The source and destination IP address of the traffic, or
- The source and destination MAC address of the traffic.

How FAT PWs distribute flows over ECMPs and bundle links

FAT PWs are designed to improve traffic distribution across ECMPs and link bundles by introducing greater entropy through flow labels. This ensures efficient utilization of network resources and enhances performance.

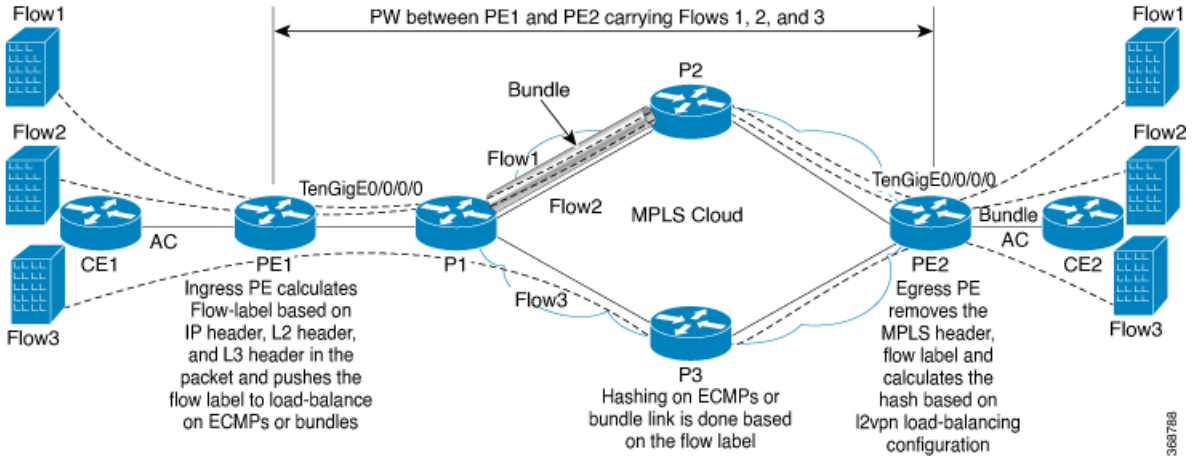
Summary

The process of distributing flows over Equal-Cost Multi-Paths (ECMPs) and bundle links in a FAT Pseudowire (PW) involves the use of flow labels to enhance traffic load balancing.

The key components involved in the process are:

- Flow label: A unique identifier derived from source and destination MAC and IP addresses, inserted into the label stack.
- Ingress PE (PE1):Adds the flow label to the traffic.
- Egress PE (PE2): Discards the flow label and uses a single EVPN label for unicast traffic.
- Core routers (P routers): Use the flow label for load balancing.

Workflow



These stages describe how flow-based load balancing operates to optimize traffic distribution across PE device:

1. Flow label generation:

- The ingress PE (PE1) generates a flow label for each unique incoming flow based on source and destination MAC and IP addresses.
- The flow label is inserted after the VC label and before the control word (if any).

2. Traffic forwarding:
   - PE1 forwards the traffic with the flow label to the core routers.
   - Core routers use the flow label, along with other information like MAC and IP addresses, to perform load balancing across ECMPs and bundle links.
3. Traffic handling at egress PE:
   - The egress PE (PE2) receives the traffic and discards the flow label.
   - PE2 uses a single EVPN label for all unicast traffic, ensuring compatibility with mixed traffic types (with or without flow labels).

The process enables efficient load balancing of traffic across ECMPs and bundle links, improving network performance and resource.

Restrictions for configuring flow label for EVPN E-Line

To ensure proper configuration and avoid unsupported scenarios, follow these restrictions:

- Unsupported services
  - This feature is not supported for EVPN Point-to-Multipoint (P2MP) of VPLS and Ethernet LAN (E-LAN) services.
  - This feature is not supported for EVPN flexible cross-connect service.
  - This feature is not supported for EVPN E-Line multi-homing.
- Supported configuration
  - This feature is supported only for EVPN E-Line single homing.
  - Ensure that AC bundle interfaces are configured with ESI-0 only

Failure to comply with these restrictions may result in configuration errors or unsupported network behavior.

Configure flow label for EVPN E-Line

Configure a flow label for EVPN E-Line on both PE1 and PE2 to enable load balancing and efficient traffic management.

EVPN E-Line is a point-to-point Ethernet VPN service that uses MPLS encapsulation. Configuring a flow label ensures proper load balancing across the network.

Before you begin

- Ensure that PE1 and PE2 are running compatible software versions.
- Verify that the interfaces and EVPN instances are pre-configured.

Procedure

Step 1 Configure L2VPN xconnect group.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# xconnect group evpn-vpws
Router(config-l2vpn-xc)# p2p evpn1
Router(config-l2vpn-xc-p2p)# interface TenGigE0/0/0/0
Router(config-l2vpn-xc-p2p)# neighbor evpn evi 1 target 2 source 1
Router(config-l2vpn-xc-p2p)# commit
```

Step 2 Configure the EVPN instance.

Example:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 1
Router(config-evpn-instance)# load-balancing
Router(config-evpn-instance-lb)# flow-label static
Router(config-evpn-instance-lb)# commit
```

Step 3 Flow label for EVPN E-Line running configuration.

Example:

```
l2vpn
 xconnect group evpn-vpws
  p2p evpn1
   interface TenGigE0/0/0/0
   neighbor evpn evi 1 target 2 source 1
   !
  !
evpn
 evi 1
   load-balancing
   flow-label static
  !
!
```

Step 4 Use the **show l2vpn xconnect detail** command to verify that EVPN E-Line flow label is configured.

Example:

```
Router# show l2vpn xconnect detail
Group evpn-vpws, XC evpn1, state is up; Interworking none
  AC: TenGigE0/0/0/0, state is up
    Type Ethernet
    MTU 1500; XC ID 0x1; interworking none
    Statistics:
      packets: received 21757444, sent 0
      bytes: received 18226521128, sent 0
  EVPN: neighbor 100.100.100.2, PW ID: evi 1, ac-id 2, state is up ( established )
    XC ID 0xc0000001
    Encapsulation MPLS
```

```
Encap type Ethernet, control word disabled
 Sequencing not set
 LSP : Up
 Flow Label flags configured (Tx=1,Rx=1) statically

   EVPN         Local                          Remote
   ------------ ------------------------------ ------------------------------
   Label        64002                          64002
   MTU          1500                           1500
   Control word disabled                       disabled
   AC ID        1                              2
   EVPN type    Ethernet                       Ethernet

   ------------ ------------------------------ ------------------------------
 Create time: 30/10/2018 03:04:16 (00:00:40 ago)
 Last time status changed: 30/10/2018 03:04:16 (00:00:40 ago)
 Statistics:
   packets: received 0, sent 21757444
   bytes: received 0, sent 18226521128
```

Private line emulation over EVPN E-Line single-homing

A Packet Label Edge (PLE) service is a mechanism that

- enables transparent packet transfer across different port modes over MPLS networks
- supports EVPN E-Line single-homing service for PLE client traffic, and
- facilitates pseudowire channel setup between endpoints during L2VPN cross-connect establishment.

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Private Line Emulation over EVPN-VPWS Single Homed	Release 24.4.1	Introduced in this release on: Fixed Systems (8700) (select variants only*) * The Private Line Emulation over EVPN-VPWS Single Homed functionality is now extended to the Cisco 8712-MOD-M routers.

Feature Name	Release Information	Feature Description
Private Line Emulation over EVPN-VPWS Single Homed	Release 24.3.1	Introduced in this release on: Fixed Systems (8200, 8700); Modular Systems (8800 [LC ASIC: P100]) (select variants only*) * The Private Line Emulation over EVPN-VPWS Single Homed functionality is now extended to: • 8212-48FH-M • 8711-32FH-M • 88-LC1-52Y8H-EM • 88-LC1-12TH24FH-E
Private Line Emulation over EVPN-VPWS Single Homed	Release 24.2.11	Introduced in this release on: Modular Systems (8800 [LC ASIC: P100])(select variants only*) * The Private Line Emulation over EVPN-VPWS Single Homed functionality is now extended to routers with the 88-LC1-36EH line cards.

Feature Name	Release Information	Feature Description
Private Line Emulation over EVPN-VPWS Single Homed	Release 7.11.1	Introduced in this release on: Cisco 8011-2X2XP4L PLE Service Endpoint Router. You can now configure EVPN VPWS to carry the client traffic from ports like FC, OTN, SDH, SONET, or Ethernet and forward the traffic to the core network by using Private Line Emulation (PLE). PLE emulates the switching capabilities of FC, OTN, SDH, SONET, or Ethernet ports without needing a dedicated equipment and allows interconnecting optical networks with Ethernet networks. This feature introduces the **port-mode** command. This release introduces new and modified YANG data models for PLE. For the list of supported data models, see *Supported Yang Data Models for PLE*. You can access these data models from the Github repository.

Key components and processes of PLE

Packet Line Emulation (PLE) service is a mechanism designed to enable the seamless transfer of packets from various port modes over MPLS networks. PLE integrates the use of EVPN-VPWS, BGP, pseudowires, and CEM to provide a robust mechanism for transparent packet transfer across MPLS networks, while maintaining compatibility with native client interfaces. This is achieved through the following key components and processes:

- Traffic handling: PLE client traffic is carried using an EVPN-VPWS single-homed setup. The traffic flow begins at the PLE initiator and terminates at the PLE endpoint.
- BGP session for route exchange: PLE endpoints establish a BGP session to exchange EVPN routing information, ensuring proper connectivity and signaling between endpoints.
- Pseudowire setup: A pseudowire channel is established between endpoints when a Layer 2 VPN cross-connect is configured. This cross-connect links the PLE client, represented as a Circuit Emulation (CEM) interface, to the remote node.
- Circuit Emulation (CEM): CEM provides native client interfaces and allows data to be transmitted over Ethernet or MPLS networks. It encapsulates client bitstreams into packet payloads with pseudowire emulation headers, enabling the transport of circuits over a Packet Switched Network (PSN).

- Encapsulation and transport: PLE client traffic is encapsulated by the PLE initiator and carried over EVPN-VPWS Layer 2 services running on Segment Routing or MPLS tunnels. Label imposition and disposition facilitate traffic flow between the client and core networks.
- Traffic extraction: At the PLE terminator node, the encapsulated bitstreams are extracted from EVPN packets and delivered to the PLE client interface. This process is governed by the client attribute and CEM profile, ensuring accurate delivery.

These stages are crucial for understanding traffic flow and the mechanisms of encapsulation and decapsulation, which are employed to emulate traditional private line services within an EVPN:

Restrictions for PLE over EVPN E-Line single-homing

These restrictions apply exclusively to the Cisco 8011-2X2XP4L PLE Service Endpoint Router running IOS XR Release 7.11.1:

- Load balancing: PLE traffic in the core does not support load balancing. This limitation arises because PLE is incompatible with ECMP or core bundles containing more than one member link.
- Software offloading: Supported only for SR-TE performance monitoring. Consequently, Fast Reroute (FRR) convergence is not achievable.

For more information on label stacking, see *MPLS Configuration Guide for Cisco 8000 Series Routers*.

Supported hardware for PLE

PLE is supported on Cisco 8011-2X2XP4L PLE Service Endpoint Router with SFP+ optical transceivers and supports these port mode options:

- Ethernet – 10GE
- Fiber channel (FC) – 1G, 2G, 4G, 8G, 16G, and 32G
- Optical Transport Network (OTN) – OTU2 and OTU2e
- Synchronous Digital Hierarchy (SDH) – STM16 and STM64
- Synchronous Optical Networking (SONET) – OC48 and OC192

PLE forwarding flow - how MPLS label imposition works

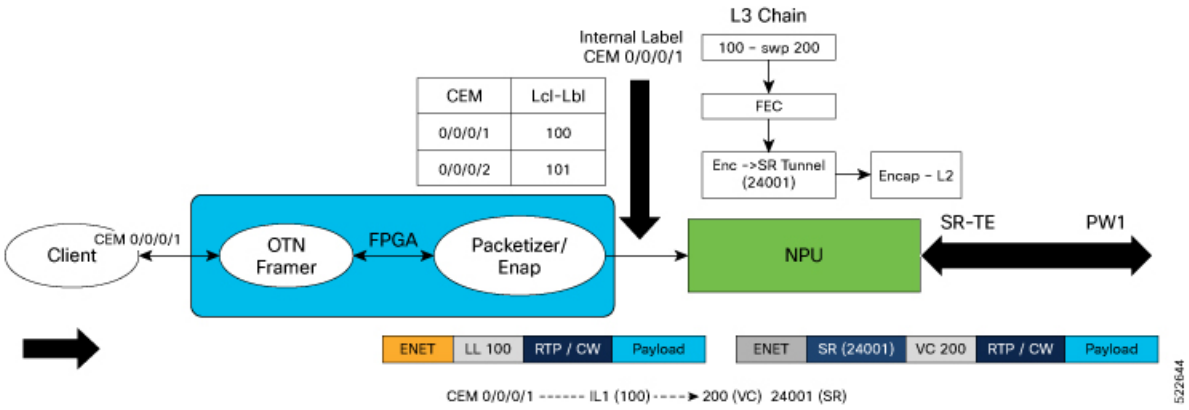
Summary

The MPLS label imposition process involves adding an MPLS label to a data packet to enable its forwarding within an MPLS network. The key components involved in this process are:

- PE router: Adds an MPLS label to packets entering the MPLS network.
- Field Programmable Gate Array (FPGA): Acts as a forwarding block, assigning an internal local label to traffic from the client.
- Network Processing Unit (NPU): Replaces the internal local label with a Virtual Circuit (VC) label and forwards the traffic using a transport label.

Workflow

Figure 1: PLE forwarding flow – imposition



The MPLS label imposition process involves these stages:

1. Traffic reception
    - Traffic from the client, which can be in various port modes, such as, FC, OTN, SDH, SONET, Ethernet, is received by the FPGA.
    - The FPGA assigns an internal local label, for example, 100 to the traffic and forwards it to the NPU.
2. Label replacement
    - The NPU receives the traffic with the internal local label from the FPGA.
    - In the forwarding L3 chain, the NPU replaces the internal local label, for example, 100, with a Virtual Circuit (VC) label, for example 200, also known as the PW label.
3. Traffic forwarding
    - The NPU adds a transport label, for example, 24001, to the packet.
    - The traffic is forwarded towards the core network using the transport label.

    The MPLS label imposition process ensures that packets are correctly labeled and forwarded within the MPLS network, enabling efficient and seamless data transmission.

PLE forwarding flow - how MPLS label disposition works

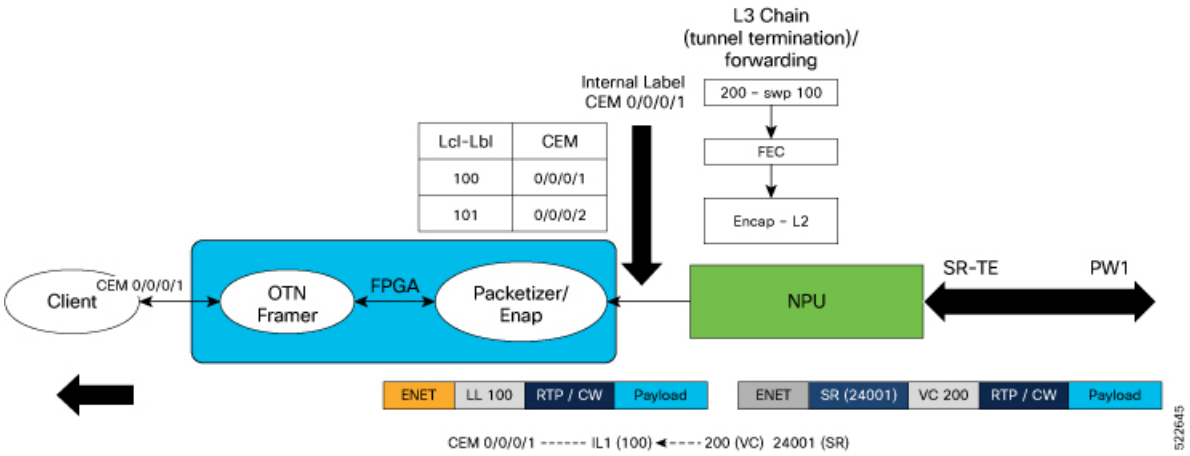
Summary

The PLE forwarding flow involves the disposition of MPLS labels to forward traffic from the core network to the client network. This process ensures efficient traffic delivery by removing MPLS labels and forwarding packets to the client. The key components involved in the process are:

- PE router: Receives MPLS packets, makes forwarding decisions, and removes MPLS labels.
- NPU: Determines the outgoing interface based on VC label allocation.
- FPGA: Maps internal local labels to the appropriate CEM interface for client delivery.

Workflow

Figure 2: PLE forwarding flow – disposition



The process involves these stages:

1. Traffic reception: The NPU receives traffic with a VC label.
2. Outgoing interface determination: The NPU determines the outgoing interface based on the VC label allocation.
3. Label replacement: The VC label, for example, 200, is replaced with an internal local label, for example, 100, and sent to the FPGA.
4. Label mapping: The FPGA maps the internal local label to the CEM interface, example 0/0/0/1.
5. Traffic forwarding: The traffic is forwarded to the client through the mapped CEM interface.

   The PLE forwarding flow ensures that traffic is efficiently delivered from the core network to the client network by removing MPLS labels and forwarding packets to the appropriate client interface.

PLE transport mechanism

Summary

The PLE forwarding flow involves the disposition of MPLS labels to forward traffic from the core network to the client network. This process ensures efficient traffic delivery by removing MPLS labels and forwarding packets to the client. The key components involved in the process are:

- Circuit-style SR-TE policies: Configured statically as preferred paths within pseudowire classes.
- Pseudowire class: Associates SR-TE policies with working or protected pseudowires.
- Co-router bidirectional paths: Ensures traffic flows in both directions with guaranteed latency.

Workflow

The process involves these stages:

1. Policy configuration: Circuit-style SR-TE policies are configured statically as preferred paths within pseudowire classes.

2. Pseudowire association: An SR-TE policy is associated per pseudowire by assigning the corresponding pseudowire class.
3. Traffic transport: The configured SR-TE policies ensure guaranteed latency, bandwidth, and end-to-end path protection for client traffic.

   The PLE transport mechanism provides a reliable and efficient method for transporting client traffic over networks, ensuring guaranteed performance and protection.

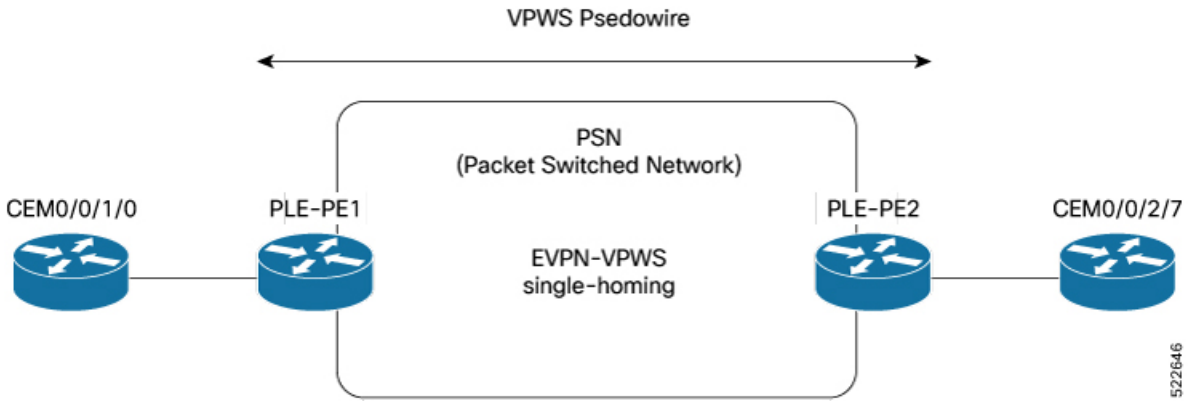
For more information on SR-TE policies, see the *Configure SR-TE Policies* section in the *Segment Routing Configuration Guide for Cisco 8000 Series Routers*.

Configure PLE over EVPN E-Line

Configure PLE over EVPN E-Line to enable efficient traffic management and synchronization

This task involves setting up PLE over EVPN E-Line with prerequisites, topology setup, and configuration steps.

Perform these tasks to configure EVPN-VPWS over SR-TE policy with explicit path. For more information on SR-TE policies, see the *Configure SR-TE Policies* section in the *Segment Routing Configuration Guide for Cisco 8000 Series Routers, IOS XR*.



In this topology, CEM interfaces are connected to PLE interfaces. The PLE interfaces, PE1 and PE2, are connected through EVPN-VPWS single homing. The PLE interface can be: Ethernet, OTN, FC, or SONET/SDH.

Before you begin

- Install all mandatory Cisco RPMs, such as RSVP for MPLS-TE.

  For more information, see the *Implementing RSVP for MPLS-TE* section in the *MPLS Configuration Guide for Cisco 8000 Series Routers*.
- Ensure clock synchronization between routers using SyncE or PTP to prevent data traffic drops.
- Verify that the core interface bandwidth exceeds the access interface bandwidth. For example, if CE traffic is 10G, the core interface must support at least 25G.

Procedure

Step 1 Enable frequency synchronization.

Example:

```
/* Enable frequency synchronization on PLE-PE1 */
Router(config)# frequency synchronization
Router(config-freqsync)# quality itu-t option 1
Router(config-freqsync)# exit
Router(config)# interface TwentyFiveGigE0/0/0/24
Router(config-if)# frequency synchronization
Router(config-if-freqsync)# quality transmit exact itu-t option 1 PRC
```

SyncE or PTP must be UP. Use the **show frequency synchronization interfaces** command to verify that the clock is transmitted.

Example:

```
/* Enable frequency synchronization on PLE-PE2 */
Router(config)# frequency synchronization
Router(config-freqsync)# quality itu-t option 1
Router(config-freqsync)# exit
Router(config)# interface TwentyFiveGigE0/0/0/32
Router(config-if)# frequency synchronization
Router(config-if-freqsync)# selection input
Router(config-if-freqsync)# priority 1
Router(config-if-freqsync)# wait-to-restore 0
```

Use the **show frequency synchronization selection** command to verify if PLE-PE2 is LOCKED to PLE-PE1's clock.

Step 2 Bring up the optics controller in CEM packet mode.

Configure the optics controller based on the port mode type (Ethernet, OTN, FC, or SONET/SDH). The examples show port mode configuration for all the types of port modes. Use the relevant command according to the port mode type of the PLE interface.

Example:

Bring up the optics controller in CEM packet mode with appropriate speed on PLE-PE1.

Ethernet:

```
Router(config)# controller Optics0/0/1/0
Router(config-Optics)# port-mode Ethernet framing cem-packetize rate 10GE
Router(config-Optics)# exit
Router(config)# controller Optics0/0/1/5
Router(config-Optics)# port-mode Ethernet framing cem-packetize rate 1GE
```

OTN:

```
Router(config)# controller Optics0/0/2/0
Router(config-Optics)# port-mode otn framing cem-packetize rate otu2
Router(config-Optics)# exit
Router(config)# controller Optics0/0/2/0
Router(config-Optics)# port-mode otn framing cem-packetize rate otu2e
```

Fiber Channel:

```
Router(config)# controller Optics0/0/1/6
Router(config-Optics)# port-mode FC framing cem-packetize rate FC1
```

Note
Port mode FC32 is supported only on the even ports (Port 0, 2, 4, and 6) of the MPA.

SONET/SDH:

```
Router(config)# controller optics 0/0/2/4
Router(config-Optics)# port-mode sonet framing cem-packetize rate OC48
Router(config-Optics)# exit
Router(config)# controller optics 0/0/2/5
Router(config-Optics)# port-mode sdh framing cem-packetize rate STM16
```

Bring up the optics controller in CEM packet mode with appropriate speed on PLE-PE2.

Ethernet:

```
Router(config)# controller Optics0/0/2/7
Router(config-Optics)# port-mode Ethernet framing cem-packetize rate 10GE
Router(config-Optics)# exit
Router(config)# controller Optics0/0/1/5
Router(config-Optics)# port-mode Ethernet framing cem-packetize rate 1GE
```

OTN:

```
Router(config)# controller Optics0/0/2/0
Router(config-Optics)# port-mode otn framing cem-packetize rate otu2
Router(config-Optics)# exit
Router(config)# controller Optics0/0/2/0
Router(config-Optics)# port-mode otn framing cem-packetize rate otu2e
!
```

Fiber Channel:

```
Router(config)# controller Optics0/0/1/6
Router(config-Optics)# port-mode FC framing cem-packetize rate FC1
```

Note
Port mode FC32 is supported only on the even ports (Port 0, 2, 4, and 6) of the MPA.

SONET/SDH:

```
Router(config)# controller optics 0/0/2/4
Router(config-Optics)# port-mode sonet framing cem-packetize rate OC48
Router(config-Optics)# exit
Router(config)# controller optics 0/0/2/5
Router(config-Optics)# port-mode sdh framing cem-packetize rate STM16
```

Step 3 Configure access and core interfaces.

Example:

Configure the access interface for the client and then the core interface.

Configure the access and core interfaces on PLE-PE1.

Access interface: Repeat this for each port mode configuration.

```
Router(config)# interface CEM0/0/1/0
Router(config-if)# l2transport
```

Core interface:

```
Router(config)# interface TwentyFiveGigE0/0/0/24
Router(config-if)# ipv4 address 14.1.0.1 255.255.255.252
```

Example:

Configure the access and core interfaces on PLE-PE2.

Access interface: Repeat this for each port mode configuration.

```
Router(config)# interface CEM0/0/2/7
Router(config-if)# l2transport
```

Core interface:

```
Router(config)# interface TwentyFiveGigE0/0/0/32
Router(config-if)# ipv4 address 14.1.0.2 255.255.255.252
```

Step 4 Configure loopback interface to establish BGP-EVPN neighborship.

Example:

```
/* Configure loopback interface on PLE-PE1 */
Router(config)# interface Loopback0
Router(config-if)# ipv4 address 1.1.1.1 255.255.255.255

/* Configure loopback interface on PLE-PE2 */
Router(config)# interface Loopback0
Router(config-if)# ipv4 address 1.1.1.4 255.255.255.255
!
```

Step 5 Configure IS-IS IGP.

Example:

Configure IS-IS IGP to advertise the configured loopback and core interfaces.

Note
You cannot configure Topology-Independent Loop-Free Alternate (TI-LFA) on the links used by circuit-styled SR-TE tunnel. The adjacency SID label is unprotected for circuit-styled SR-TE, which does not support TI-LFA.

Configure IS-IS IGP on PLE-PE1.

```
Router(config)# router isis core
Router(config-isis)# is-type level-2-only
Router(config-isis)# net 49.0000.0000.0000.0001.00
Router(config-isis)# nsr
Router(config-isis)# nsf cisco
Router(config-isis)# log adjacency changes
Router(config-isis)# address-family ipv4 unicast
Router(config-isis-af)# metric-style wide
Router(config-isis-af)# segment-routing mpls sr-prefer
Router(config-isis-af)# segment-routing bundle-member-adj-sid
Router(config-isis-af)# commit
Router(config-isis-af)# exit
Router(config-isis)# interface Loopback0
```

```
Router(config-isis-if)# point-to-point
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# prefix-sid index 1
Router(config-isis-if-af)# exit
Router(config-isis)# interface TwentyFiveGigE0/0/0/24
Router(config-isis-if)# point-to-point
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# adjacency-sid absolute 28121 >>>>> Adjacency-SID must be unprotected for
 circuit-styled SR-TE
Router(config-isis-if-af)# commit
Router(config-isis-if-af)# exit
```

Configure IS-IS IGP on PLE-PE2.

```
Router(config)# router isis core
Router(config-isis)# is-type level-2-only
Router(config-isis)# net 49.0000.0000.0000.0004.00
Router(config-isis)# nsr
Router(config-isis)# nsf cisco
Router(config-isis)# log adjacency changes
Router(config-isis)# address-family ipv4 unicast
Router(config-isis-af)# metric-style wide
Router(config-isis-af)# segment-routing mpls sr-prefer
Router(config-isis-af)# segment-routing bundle-member-adj-sid
Router(config-isis-af)# commit
Router(config-isis-af)# exit
Router(config-isis)# interface Loopback0
Router(config-isis-if)# point-to-point
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# prefix-sid index 4
Router(config-isis-if-af)# exit
Router(config-isis)# interface TwentyFiveGigE0/0/0/32
Router(config-isis-if)# point-to-point
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if-af)# adjacency-sid absolute 28211 >>>>> Adjacency-SID must be unprotected for
 circuit-styled SR-TE
Router(config-isis-if-af)# commit
Router(config-isis-if-af)# exit
```

Step 6 Configure performance measurement to enable the liveness monitoring of the SR policy.

Example:

Configure performance measurement on PLE-PE1.

```
Router(config)# performance-measurement
Router(config-perf-meas)# liveness-profile sr-policy name RED
Router(config-pm-ld-srpolicy)# probe
Router(config-pm-ld-srpolicy-probe)# measurement-mode loopback
Router(config-pm-ld-srpolicy-probe)# burst-interval 3000
Router(config-pm-ld-srpolicy-probe)# exit
Router(config-pm-ld-srpolicy)# exit

Router(config-perf-meas)# liveness-profile sr-policy name BLUE
Router(config-pm-ld-srpolicy)# probe
Router(config-pm-ld-srpolicy-probe)# measurement-mode loopback
Router(config-pm-ld-srpolicy-probe)# burst-interval 30
```

Configure performance measurement on PLE-PE2.

```
Router(config)# performance-measurement
```

```
Router(config-perf-meas)# liveness-profile sr-policy name RED
Router(config-pm-ld-srpolicy)# probe
Router(config-pm-ld-srpolicy-probe)# measurement-mode loopback
Router(config-pm-ld-srpolicy-probe)# burst-interval 3000
Router(config-pm-ld-srpolicy-probe)# exit
Router(config-pm-ld-srpolicy)# exit

Router(config-perf-meas)# liveness-profile sr-policy name BLUE
Router(config-pm-ld-srpolicy)# probe
Router(config-pm-ld-srpolicy-probe)# measurement-mode loopback
Router(config-pm-ld-srpolicy-probe)# burst-interval 30
```

Step 7 Configure segment routing traffic engineering tunnels.

Example:

Configure circuit-styled SR-TE tunnels. SR-TE is supported only with explicit path specified by adjacency SID labels. The adjacency SID labels must be unprotected for circuit-styled SR-TE. This example shows configuration of explicit path between PE1 and PE2.

Configure segment routing traffic engineering tunnels on PLE-PE1.

```
Router(config)# segment-routing
Router(config-sr)# global-block 80000 111999
Router(config-sr)# local-block 25000 28999
Router(config-sr)# traffic-eng
Router(config-sr-te)# segment-list pe1-pe2-forward-path
Router(config-sr-te-sl)# index 1 mpls label 28121
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list pe1-pe2-reverse-path
Router(config-sr-te-sl)# index 1 mpls label 28211
Router(config-sr-te-sl)# exit
Router(config-sr-te)# policy pe1-pe2-circuit-styled-srte
Router(config-sr-te-policy)# color 10 end-point ipv4 1.1.1.4
Router(config-sr-te-policy)# path-protection
Router(config-sr-te-policy)# candidate-paths
Router(config-sr-te-policy-path)# preference 10
Router(config-sr-te-policy-path-pref)# explicit segment-list pe1-pe2-forward-path >>>>> Explicit
path
Router(config-sr-te-policy-path-pref)# reverse-path segment-list pe1-pe2-reverse-path
Router(config)# performance-measurement
Router(config-perf-meas)# liveness-detection
Router(config-perf-meas)# liveness-profile backup name RED
Router(config-perf-meas)# liveness-profile name BLUE
```

Configure segment routing traffic engineering tunnels on PLE-PE2.

```
Router(config)# segment-routing
Router(config-sr)# global-block 80000 111999
Router(config-sr)# local-block 25000 28999
Router(config-sr)# traffic-eng
Router(config-sr-te)# segment-list pe1-pe2-forward-path
Router(config-sr-te-sl)# index 1 mpls label 28211
Router(config-sr-te-sl)# exit
Router(config-sr-te)# segment-list pe1-pe2-reverse-path
Router(config-sr-te-sl)# index 1 mpls label 28121
Router(config-sr-te-sl)# exit
Router(config-sr-te)# policy pe1-pe2-circuit-styled-srte
Router(config-sr-te-policy)# color 10 end-point ipv4 1.1.1.1
Router(config-sr-te-policy)# path-protection
Router(config-sr-te-policy)# candidate-paths
```

```
Router(config-sr-te-policy-path)# preference 10
Router(config-sr-te-policy-path-pref)# explicit segment-list pe1-pe2-forward-path >>>>> Explicit
path
Router(config-sr-te-policy-path-pref)# reverse-path segment-list pe1-pe2-reverse-path
Router(config)# performance-measurement
Router(config-perf-meas)# liveness-detection
Router(config-perf-meas)# liveness-profile backup name RED
Router(config-perf-meas)# liveness-profile name BLUE
```

Step 8 Configure BGP EVPN neighbor session.

Example:

Configure L2VPN EVPN address family under BGP to establish a BGP-EVPN neighbor session.

Configure BGP EVPN neighbor session on PLE-PE1.

```
Router(config)# router bgp 100
Router(config-bgp)# bgp router-id 1.1.1.1
Router(config-bgp)# bgp graceful-restart
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# exit
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 1.1.1.4
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# update-source Loopback0
Router(config-bgp-nbr)# exit
Router(config-bgp)# graceful-restart
Router(config-bgp)# address-family l2vpn evpn
```

Configure BGP EVPN neighbor session on PLE-PE2.

```
Router(config)# router bgp 100
Router(config-bgp)# bgp router-id 1.1.1.4
Router(config-bgp)# bgp graceful-restart
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# exit
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 1.1.1.1
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# update-source Loopback0
Router(config-bgp-nbr)# exit
Router(config-bgp)# graceful-restart
Router(config-bgp)# address-family l2vpn evpn
```

Step 9 Configure EVPN VPWS.

Example:

Configure EVPN VPWS with PW class and xconnect service to carry the PLE client traffic.

Configure EVPN VPWS on PLE-PE1.

```
Router(config)# l2vpn
Router((config-l2vpn)# pw-class pw-cs-srte
Router((config-l2vpn-pwc)# encapsulation mpls
Router((config-l2vpn-pwc-mpls)# preferred-path sr-te policy srte_c_10_ep_1.1.1.6
Router(config)# xconnect group evpn_vpws
Router(config)# p2p p1
Router(config)# interface CEM0/0/1/0
```

```
Router(config)# neighbor evpn evi 10 target 1 source 2
Router(config)# pw-class pw-cs-srte
```

Configure EVPN VPWS on PLE-PE2.

```
Router(config)# l2vpn
Router((config-l2vpn)# pw-class pw-cs-srte
Router((config-l2vpn-pwc)# encapsulation mpls
Router((config-l2vpn-pwc-mpls)# preferred-path sr-te policy srte_c_10_ep_1.1.1.1
Router(config)# xconnect group evpn_vpws
Router(config)# p2p p1
Router(config)# interface CEM0/0/2/7
Router(config)# neighbor evpn evi 10 target 1 source 2
Router(config)# pw-class pw-cs-srte
```

Step 10 Configure QoS policy on CEM interface.

Example:

Configure QoS policy to manage congestion on PLE client traffic. In QoS for PLE, you can mark the MPLS experimental with only the topmost label and set the traffic class with only the default class.

Configure QoS policy on PLE-PE1.

Access interface configuration

```
Router(config)# policy-map ple-policy
Router(config-pmap)# class class-default
Router(config-pmap-c)# set mpls experimental topmost 7
Router(config-pmap-c)# set traffic-class 2
Router(config-pmap-c)# end-policy-map
!
Router(config)# interface CEM0/0/1/0
Router(config-if)# l2transport
Router(config-if)# service-policy input ple-policy
!
```

Core interface configuration

```
Router(config)# class-map match-any tc2
Router(config-cmap)# match traffic-class 2
Router(config-cmap)# end-class-map
!
Router(config)# policy-map core
Router(config-pmap)# class tc2
Router(config-pmap-c)# priority level 1
Router(config-pmap-c)# shape average percent 100
Router(config-pmap-c)# end-policy-map
!
Router(config)# interface TwentyFiveGigE0/0/0/24
Router(config-if)# mtu 9200
Router(config-if)# service-policy output core
Router(config-if)# ipv4 address 13.30.1.1 255.255.255.252
```

Configure QoS policy on PLE-PE2.

Access interface configuration

```
Router(config)# policy-map ple-policy
Router(config-pmap)# class class-default
Router(config-pmap-c)# set mpls experimental topmost 7
```

```
Router(config-pmap-c)# set traffic-class 2
Router(config-pmap-c)# end-policy-map
!
Router(config)# interface CEM0/0/2/7
Router(config-if)# l2transport
Router(config-if)# service-policy input ple-policy
```

Core interface configuration

```
Router(config)# class-map match-any tc2
Router(config-cmap)# match traffic-class 2
Router(config-cmap)# end-class-map
!
Router(config)# policy-map core
Router(config-pmap)# class tc2
Router(config-pmap-c)# priority level 1
Router(config-pmap-c)# shape average percent 100
Router(config-pmap-c)# end-policy-map
!
Router(config)# interface TwentyFiveGigE0/0/0/32
Router(config-if)# mtu 9200
Router(config-if)# service-policy output core
Router(config-if)# ipv4 address 46.10.1.2 255.255.255.252
```

Step 11 Use these show commands to view the configuration.

Example:

Use the **show isis neighbors** command to verify the IS-IS configuration.

```
Router# show isis neighbors
Fri Nov 12 09:04:13.638 UTC

IS-IS core neighbors:
System Id      Interface        SNPA           State Holdtime Type IETF-NSF
PLE-Core-PE2   TF0/0/0/24       *PtoP*         Up    28       L2   Capable

Total neighbor count: 1
```

```
Router# show isis segment-routing label table

Fri Nov 12 09:25:18.488 UTC

IS-IS core IS Label Table
Label         Prefix                   Interface
----------    -----------------        ---------
16001         1.1.1.1/32               Loopback0
16004         1.1.1.4/32
```

```
Router# show mpls forwarding prefix 1.1.1.4/32
Fri Nov 12 09:25:54.898 UTC
Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
16004  Pop         SR Pfx (idx 4)     TF0/0/0/24   14.1.0.2        104332
```

Use the **show performance-measurement sr-policy color 203** command to verify the performance measurement.

```
Router# show performance-measurement sr-policy color 203
Mon Mar 14 17:54:32.403 IST

------------------------------------------------------------------
0/RP0/CPU0
------------------------------------------------------------------
```

```
SR Policy name: srte_c_203_ep_1.1.1.1
Color : 203
Endpoint : 1.1.1.1
Number of candidate-paths : 1

Candidate-Path:
Instance : 8
Preference : 10
Protocol-origin : Configured
Discriminator : 10
Profile Keys:
Profile name : BLUE
Profile type : SR Policy Liveness Detection
Source address : 1.1.1.6
Number of segment-lists : 1
Liveness Detection: Enabled
Session State: Up
Last State Change Timestamp: Mar 14 2022 17:53:45.207
Missed count: 0

------------------------------------------------------------------------
0/0/CPU0
------------------------------------------------------------------------
```

Use the **show segment-routing traffic-eng policy color 10 tabular** command to verify the SR-TE configuration.

```
Router# show segment-routing traffic-eng policy color 10 tabular
Fri Nov 12 09:15:57.366 UTC

 Color                                Endpoint  Admin   Oper                  Binding
                                                State  State                     SID
------ ---------------------------------------- ------ ------ ------------------------
    10                                 1.1.1.4     up     up                   24010
```

Use the **show bgp l2vpn evpn neighbors brief** command to verify BGP EVPN neighbor session configuration.

```
Router# show bgp l2vpn evpn neighbors brief
Fri Nov 12 09:10:22.999 UTC

Neighbor        Spk    AS Description                          Up/Down  NBRState
1.1.1.4           0   100                                      15:51:52 Established
```

Use the **show l2vpn xconnect** command to verify the EVPN VPWS configuration.

```
Router# show l2vpn xconnect
Fri Nov 12 09:02:44.982 UTC
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
        SB = Standby, SR = Standby Ready, (PP) = Partially Programmed,
        LU = Local Up, RU = Remote Up, CO = Connected, (SI) = Seamless Inactive

XConnect                   Segment 1                       Segment 2
Group      Name       ST   Description            ST       Description            ST
------------------------------------------------------------------------------------
evpn_vpws  p1         UP   CE0/0/1/0              UP       EVPN 10,1,24012        UP
------------------------------------------------------------------------------------
```

Use the **show policy-map targets** to verify QoS policy is configured.

This show command displays information about interfaces on which the policy maps are applied.

```
Router# show policy-map targets
Thu Jun 16 21:47:31.407 IST
1) Policymap: ple-p1 Type: qos
Targets (applied as main policy):
```

```
CEM0/0/1/0 input
Total targets: 1

Targets (applied as child policy):
Total targets: 0

2) Policymap: core Type: qos
Targets (applied as main policy):
TwentyFiveGigE0/0/0/24
Total targets: 1

Targets (applied as child policy):
Total targets: 0
```

Use **show policy-map interface TwentyFiveGigE0/0/0/24** command to view the core interface information and to verify the traffic class (TC) mapping in CEM interface.

```
Router# show policy-map interface TwentyFiveGigE0/0/0/24
Thu Jun 16 21:37:52.915 IST
TwentyFiveGigE0/0/0/24 direction input: Service Policy is not installed
TwentyFiveGigE0/0/0/24 output: core
Class tc2
    Classification Statistics    (packets/bytes)          (rate - kbps)
        Matched           :      39654778/42113374236        6816279
        Transmitted       :      39654778/42113374236        6816279
        Total Dropped     :             0/0                  0
    Queueing Statistics
        Queue ID                     : 1370
        Taildropped(packets/bytes)   : 0/0
Class class-default
    Classification Statistics    (packets/bytes)          (rate - kbps)
        Matched           :             0/0                  0
        Transmitted       :             0/0                  0
        Total Dropped     :             0/0                  0
    Queueing Statistics
        Queue ID                     : 1368
        Taildropped(packets/bytes)   : 0/0
Policy Bag Stats time: 1655395669491    [Local Time: 06/16/22 21:37:49:491]
```

Supported Yang data models for PLE

Here is the list of new and modified Yang data models supported for PLE. You can access the data models from the Github repository.

Configuration Files - New:

- Cisco-IOS-XR-controller-fc-cfg.yang
- Cisco-IOS-XR-fibrechannelmib-cfg.yang
- Cisco-IOS-XR-interface-cem-cfg.yang
- Cisco-IOS-XR-cem-class-cfg.yang

Configuration Files - Modified:

- Cisco-IOS-XR-controller-odu-cfg.yang
- Cisco-IOS-XR-controller-otu-cfg.yang

- Cisco-IOS-XR-controller-sonet-cfg.yang
- Cisco-IOS-XR-drivers-icpe-ethernet-cfg.yang

Operational Files - New:

- Cisco-IOS-XR-controller-fc-oper.yang
- Cisco-IOS-XR-interface-cem-oper.yang

Operational Files - Modified:

- Cisco-IOS-XR-controller-odu-oper.yang
- Cisco-IOS-XR-controller-otu-oper.yang