Configure EVPN IRB

This chapter introduces you to Ethernet VPN (EVPN) Integrated Routing and Bridging (IRB) feature and describe how you can configure the EVPN IRB feature.

EVPN IRB

From Release 25.4.1, you can configure EVPN IRB over an Segment Routing over IPv6 (SRv6) core.

Table 1: Feature History Table

Feature Name	Release Information	Feature Description

EVPN IRB over SRv6 core	Release 25.4.1	EVPN IRB enhances network flexibility by enabling seamless Layer 3 connectivity between hosts on different subnets over an SRv6 network. This feature allows Layer 3 forwarding among hosts across IP subnets, maintains EVPN's multi-homing capabilities, and facilitates communication between EVPN hosts or subnets and IP VPNs. Leveraging SRv6's programmable and flexible transport, this solution streamlines the integration and management of modern, diverse network environments.
EVPN IRB over MPLS core	Release 3.7.2	EVPN IRB enhances network flexibility by enabling seamless Layer 3 connectivity between hosts on different subnets over an MPLS or IP network. This feature allows Layer 3 forwarding among hosts across IP subnets, maintains EVPN's multi-homing capabilities, and supports communication between EVPN hosts or subnets and IP VPNs, making it easier to connect and manage diverse network environments.

The EVPN IRB feature is a component that

- enables L3 forwarding among hosts across different IP subnets,
- while maintaining the multi-homing capabilities of EVPN.
- Additionally, it allows EVPN hosts or subnets to communicate with IP VPNs, enhancing network flexibility and connectivity.

EVPN IRB components

To implement EVPN IRB, the network uses these key components to ensure effective traffic management and routing:

- **BGP (Border Gateway Protocol)**: It advertises subnet and host routes to the EVPN core using route-type 5 and route-type 2 messages.
- **EVPN**: It manages ethernet segment configurations and handles host route advertisements and failover scenarios.
- **L2RIB (Layer 2 Routing Information Base)**: It handles MAC or IP mobility, resolves routes, computes the best routes, and handles duplicate host detection.
- **BVI MA (Bridge Virtual Interface Manager)**: It manages IRB interfaces and supports routing by advertising the BVI's subnet and MAC addresses.
- **L2FIB (Layer 2 Forwarding Information Base)**: It ensures correct forwarding based on MAC addresses.

EVPN IRB environments

The EVPN IRB supports these environments:

- **Single-homing interface**: Customer Edge (CE) devices connect directly to a single Physical Edge (PE) router.
- **Multi-homing interface**: CE device connect to multiple PE routers through dual links, a Link Aggregation Group (LAG), or a switch.
- **Anycast gateway or Bridge Virtual Interface (BVI)**: BVI interfaces use the same IP and MAC addresses on all PE routers, enabling devices in the network to reach the gateway using the same address, regardless of which router is the designated forwarder.

EVPN-IRB core and access options

EVPN-IRB supports these core and access options:

- **Core options**:
  - **Host routing**: Enable or disable host routing to manage subnet and host route advertisements.
  - **VRF**: Integrate with VRF for seamless route management.
- **Access options**:
  - **Switch devices**: Supports switch devices with or without LAG.
  - **Y-cable configurations**: Use Y-cable configurations for redundancy.
  - **Mixed environments connections**: Operate in mixed environments with single-homing, single-active multi-homing, and all-active multi-homing connections.

Limitations and restrictions for EVPN IRB

These are the limitations and restrictions for EVPN IRB:

- SRv6 EVPN-IRB do not support Single-Flow-Active (SFA) load balalcing mode.

EVPN single-homing access EVPN gateway

EVPN single-homing access EVPN gateway is a network architecture that leverages EVPN technology to provide L2 and L3 VPN services. This setup is particularly useful for connecting CE devices to a service provider's network in a single-homed configuration by connecting each CE device to only one PE device.

Table 2: Feature History Table

Feature Name	Release Information	Feature Description

EVPN single-homing access EVPN gateway over SRv6 core	Release 25.4.1	You can deploy an EVPN single-homing access EVPN gateway to provide Layer 2 and Layer 3 VPN services using EVPN technology over an SRv6 core. In this architecture, each customer edge (CE) device is connected to only one provider edge (PE) device, enabling a single-homed configuration that simplifies connectivity between CE devices and the service provider's programmable, flexible SRv6 network.
EVPN single-homing access EVPN gateway over MPLS core	Release 3.7.2	You can deploy an EVPN single-homing access EVPN gateway to provide Layer 2 and Layer 3 VPN services using EVPN technology over a MPLS core. In this architecture, you can connect each customer edge (CE) device to only one provider edge (PE) device, enabling a single-homed configuration that simplifies connectivity between the CE devices and the service provider's network.

How EVPN single-homing access EVPN gateway works

Summary

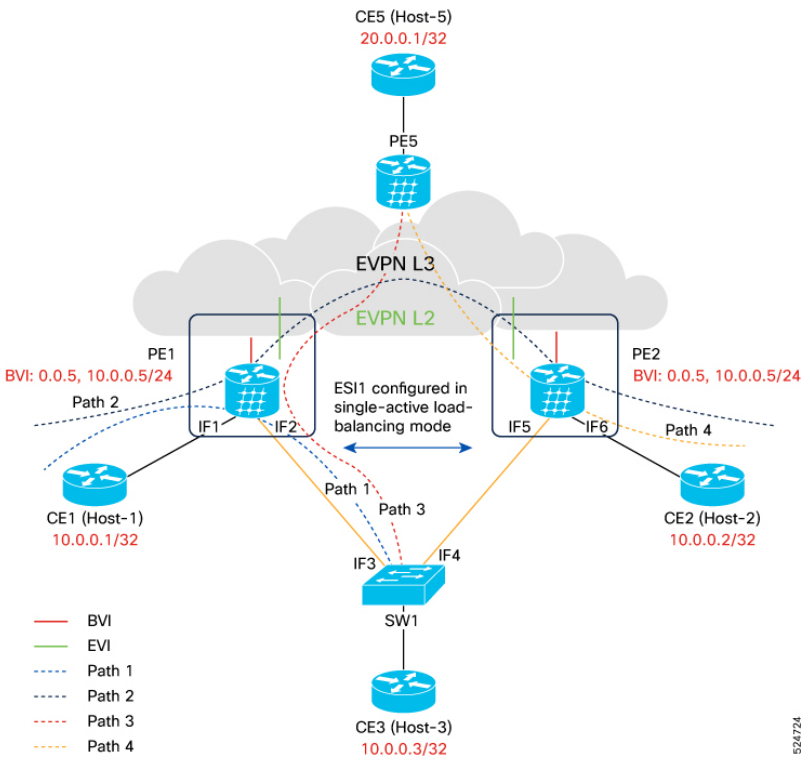
EVPN IRB single-homing workflow includes these key components:

- **CE or host routers**:
  - CE1 or Host-1 with IP address 10.0.0.1/32
  - CE2 or Host-2 with IP address 10.0.0.2/32
  - CE3 or Host-3 with IP address 10.0.0.3/32
  - CE5 or Host-5 with IP address 20.0.0.1/32
- **PE routers**:
  - PE1 with BVI IP address 10.0.0.5/24 and BVI mac address 0.0.5
  - PE2 with BVI IP address 10.0.0.5/24 and BVI mac address 0.0.5
  - PE5
- **Interfaces**:
  - IF1: Connects Host-1 to PE1
  - IF2: Connects Host-3 to PE1
  - IF3: Connects PE1 to Host-3
  - IF4: Connects PE2 to Host-3
  - IF5: Connects Host-3 to PE2
  - IF6: Connects Host-2 to PE2

- **Paths**:
  - Path 1: Host-1 communicating with Host-3 using a EVPN single-homing access EVPN gateway interface
  - Path 2: Host-1 communicating with Host-2 using a EVPN single-homing access EVPN gateway interface
  - Path 3: Host-3 communicating with Host-5 using a EVPN single-active multihoming interface
  - Path 4: Host-2 communicating with Host-5 using a EVPN single-active multihoming interface
- Switch device: SW1, connects Host-3 to PE1 and PE2
- Host routing is enabled on PE1 and PE2.
- IRB interfaces are configured as anycast.
- Host-1 and Host-2 connect to PE1 and PE2 using a EVPN single-homing interfaces.
- Host-3 connect to PE1 and PE2 using a EVPN multihoming interfaces.
- IOS XR Software performs the Designated Forwarder (DF) election for the shared Ethernet Segment Identifier (ESI) and elects IF2 as DF and IF5 as Non-Designated Forwarder (NDF).
- Interface IF5 is set to blocked state which blocks both BUM and unicast traffic.

Workflow

Figure 1: EVPN IRB single-homing network topology



Let's consider these scenarios:

- Where Host-1 wants to communicate with Host-2, which are in same subnet.
  1. Host-1 sends an ARP request, which is received by the Bridge Domain (BD) on PE1.
  2. PE1 learn the Host-1 MAC and IP addresses from these ARP packets and uses the information to program forwarding adjacencies.
  3. PE1 advertises Host-1 route using EVPN route type-2 to remote PEs. Remote PEs, such as PE2, import and install this route as a remote route.
  4. Since PE1 forwards BUM packets into the bridge domain across EVPN, Host-2 receives the ARP request and responds with a unicast ARP response.
  5. The ARP process ensures that PE2 learns Host-2 IP address 10.0.0.2/32.
  6. After Host-2 sends an unicast ARP, PE2 performs a MAC lookup for Host-1 and forwards packet to peering PE1, enabling communication between Host-2 and Host-1.
- Where Host-1 wants to communicate with Host-3, which are in same subnet.
  1. Host-1 sends an ARP request, which is received by the BD on PE1.
  2. PE1 learns Host-1's MAC and IP addresses from the ARP packets and uses this information to program forwarding adjacencies.
  3. PE1 forwards the ARP request as a broadcast within the bridge domain, allowing Host-3, which is directly connected to PE1, to receive the ARP request and respond with a unicast ARP reply.
  4. The ARP process ensures that PE1 learns Host-3's IP address 10.0.0.3/32. Since the unicast ARP reply is received only by PE1, the other PE learns the ARP entry via BGP EVPN route advertisement, enabling communication between Host-1 and Host-3.

EVPN multi-homing active-active

EVPN multi-homing access gateway enables redundant network connectivity by allowing a CE device to connect to more than one PE devices. Disruptions to the network connectivity are prevented by allowing a CE device to be connected to a PE device or several PE devices through multi-homing. Ethernet segment is the bunch of Ethernet links through which a CE device is connected to more than one PE devices.

Table 3: Feature History Table

Feature Name	Release Information	Feature Description
EVPN multi-homing active-active over SRv6 core	Release 25.4.1	You can enable an EVPN multi-homing access gateway to provide redundant connectivity for CE devices by connecting them to multiple PE devices over an SRv6 core. This approach ensures high availability and seamless Layer 2 and Layer 3 services in EVPN IRB networks, supporting both VPNv4 and VPNv6 address families and efficient host route distribution across data centers over a programmable, flexible SRv6 network.

EVPN multi-homing active-active over MPLS core	Release 3.7.2	You can enable EVPN multi-homing access gateway to provide redundant connectivity for CE devices by connecting them to multiple PE devices over MPLS core. This approach ensures high availability and seamless Layer 2 and Layer 3 services in EVPN IRB networks, supporting both VPNv4 and VPNv6 address families and efficient host route distribution across data centers.

In EVPN IRB, both EVPN and IP VPN (both VPNv4 and VPNv6) address families are enabled between ASR 9000 Data Center Interconnect (DCI) gateways. When Layer 2 (L2) stretch is not available in multiple datacenters (DC), routing is established through VPNv4 or VPNv6 routes. When Layer 2 stretch is available, host routing is applied where IP-MAC routes are learnt by ARP/IPv6 ND and are distributed to EVPN/BGP. In remote peer gateway, these IP-MAC EVPN routes are imported into IP VPN routing table from EVPN route-type 2 routes with secondary label and Layer 3 VRF route-target.

How EVPN multi-homing active-active works

Summary

EVPN IRB multi-homing active-active workflow includes these key components:

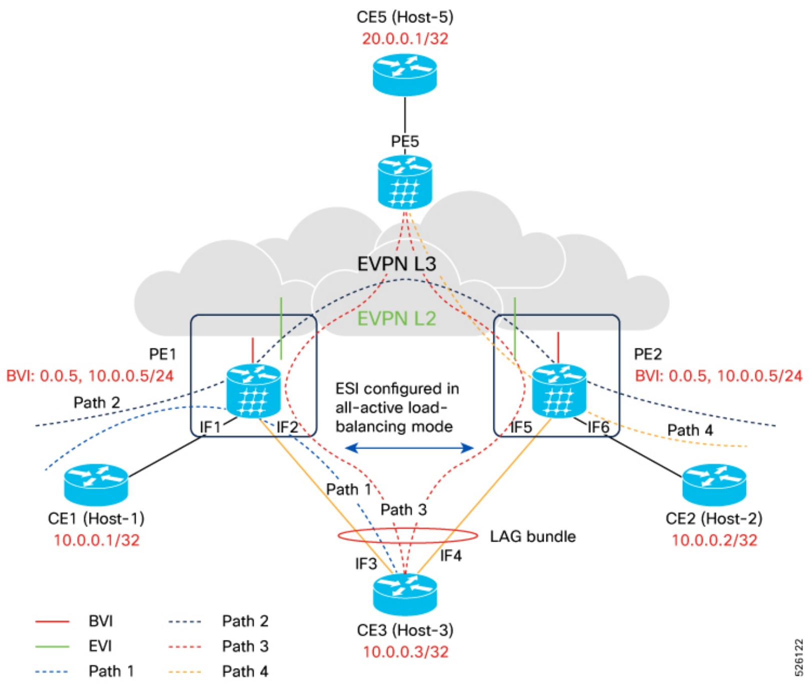
- **CE or host routers**:
  - CE1 or Host-1 with IP address 10.0.0.1/32
  - CE2 or Host-2 with IP address 10.0.0.2/32
  - CE3 or Host-3 with IP address 10.0.0.3/32
  - CE5 or Host-5 with IP address 20.0.0.1/32
- **PE routers**:
  - PE1 with BVI IP address 10.0.0.5/24 and BVI mac address 0.0.5
  - PE2 with BVI IP address 10.0.0.5/24 and BVI mac address 0.0.5
  - PE5
- **Interfaces**:
  - IF1: Connects Host-1 to PE1
  - IF2: Connects Host-3 to PE1
  - IF3: Connects PE1 to Host-3
  - IF4: Connects PE2 to Host-3
  - IF5: Connects Host-3 to PE2
- **Paths**:
  - Path 1: Host-1 communicating with Host-2 using a EVPN single-homing access EVPN gateway interface

- Path 2: Host-1 communicating with Host-3 using a EVPN single-homing access EVPN gateway interface
- Path 3: Host-3 communicating with Host-5 using a EVPN single-active multihoming interface
- Path 4: Host-2 communicating with Host-5 using a EVPN single-active multihoming interface

- Host routing is enabled on PE1 and PE2.
- IRB interfaces are configured as anycast.
- Host-1 and Host-2 connect to PE1 and PE2 using a EVPN single-homing interfaces.
- Host-3 connect to PE1 and PE2 using a EVPN multihoming interfaces.
- IOS XR Software performs the Designated Forwarder (DF) election for the shared Ethernet Segment Identifier (ESI) and elects IF2 as DF and IF5 as Non-Designated Forwarder (NDF).
- Interface IF5 is set to blocked state which blocks both BUM and unicast traffic.

Workflow

Figure 2: EVPN IRB multi-homing network topology



Let's consider these scenarios:

- Where Host-2 wants to communicate with Host-5, which are in different subnet.

  1. Host-2 send an ARP request to its gateway which is IRB interface. It basically ARPs the BVI IP address 10.0.0.5.
  2. PE2 learn the Host-2 MAC and IP addresses from these ARP packets and uses this information to program the forwarding adjacency.

3. The BVI interface on PE2 sends an ARP response to Host-2 using its BVI IP address 10.0.0.5 and MAC address 0.0.5.
4. PE2 advertises Host-2 route using EVPN route type-2 to remote PEs. Remote PEs, such as PE5, import and install this route as a remote route.
5. Since Host-5 is directly connected to PE5, it receives the ARP request and responds with a unicast ARP response.
6. The ARP process ensures that PE5 learns Host-5 IP address 20.0.0.1/32, enabling communication between Host-2 and Host-5.
7. If PE2 doesn’t have Host-5 specific route, it may use an EVPN route type-5 to forward traffic to PE5, where ARP resolves Host-5, enabling Host-2 and Host-5 to communicate.

- Where Host-5 sends a packet to Host-2. If Host-2 hasn't communicated yet, PE5 might not have Host-2 specific route.
    - If PE5 directs traffic to PE1 first, a Generalized Learning (G-LEAN) adjacency process occurs, and traffic is dropped until it is resolved.
    - Once PE5 identifies PE2 as the best destination for Host-2, it forwards the packet to PE2, and PE2 performs these steps:
        1. PE2 performs an IP lookup, finding the BVI interface as the destination.
        2. Destination MAC is set to Host-2 MAC as learned by ARP and source MAC remains as the BVI MAC address 0.0.5.
        3. PE2 performs a MAC lookup within the bridge domain and forwards the packet to Host-2.

EVPN single-active multihoming for anycast gateway IRB

The EVPN single-active multihoming for anycast gateway IRB feature supports single-active redundancy mode. In this mode, the PE nodes locally connected to an Ethernet Segment load balance traffic to and from the Ethernet Segment based on EVPN service instance (EVI). Within an EVPN service instance, only one PE forwards traffic to and from the Ethernet Segment (ES). This feature supports intersubnet scenario only.

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
EVPN single-active multihoming for anycast gateway IRB over SRv6 core	Release 25.4.1	You enable EVPN single-active multihoming for anycast gateway IRB over an SRv6 core to provide single-active redundancy, where only one PE forwards traffic for an Ethernet Segment within each EVPN service instance. This feature supports intersubnet scenarios and balances traffic based on the EVI, leveraging a programmable, flexible SRv6 network.

EVPN single-active multihoming for anycast gateway IRB over MPLS core	Release 7.11.1	You enable EVPN single-active multihoming for anycast gateway IRB over a MPLS core to provide single-active redundancy, where only one PE forwards traffic for an Ethernet Segment within each EVPN service instance. This feature supports intersubnet scenarios and balances traffic based on the EVI.

How EVPN single-active multihoming works

Summary

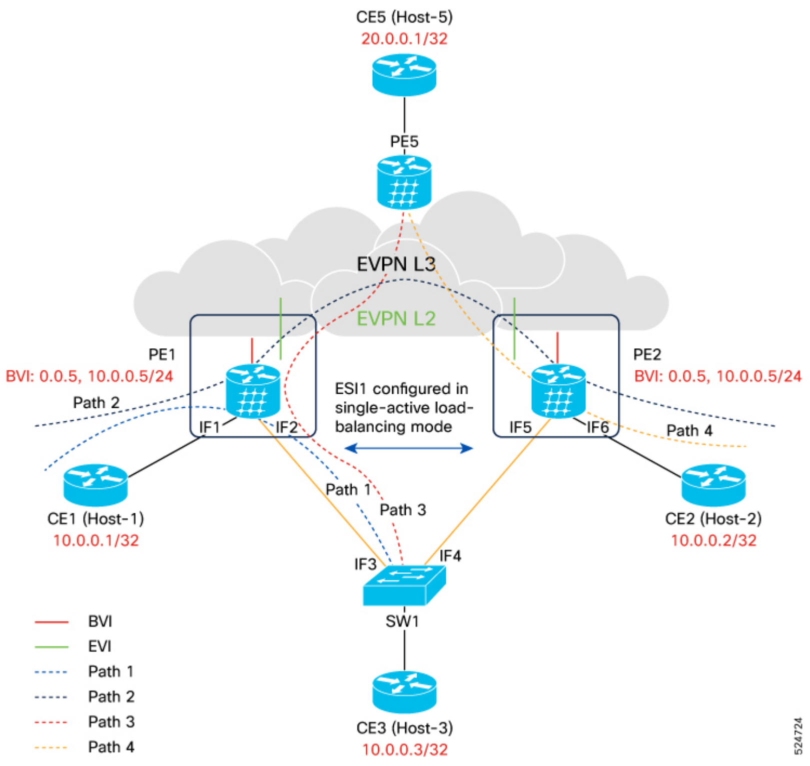
EVPN IRB single-active multihoming workflow includes these key components:

- **CE or host routers**:
  - CE1 or Host-1 with IP address 10.0.0.1/32
  - CE2 or Host-2 with IP address 10.0.0.2/32
  - CE3 or Host-3 with IP address 10.0.0.3/32
  - CE5 or Host-5 with IP address 20.0.0.1/32
- **PE routers**:
  - PE1 with BVI IP address 10.0.0.5/24 and BVI mac address 0.0.5
  - PE2 with BVI IP address 10.0.0.5/24 and BVI mac address 0.0.5
  - PE5
- **Interfaces**:
  - IF1: Connects Host-1 to PE1
  - IF2: Connects Host-3 to PE1
  - IF3: Connects PE1 to Host-3
  - IF4: Connects PE2 to Host-3
  - IF5: Connects Host-3 to PE2
- **Paths**:
  - Path 1: Host-1 communicating with Host-2 using a EVPN single-homing access EVPN gateway interface
  - Path 2: Host-1 communicating with Host-3 using a EVPN single-homing access EVPN gateway interface
  - Path 3: Host-3 communicating with Host-5 using a EVPN single-active multihoming interface
  - Path 4: Host-2 communicating with Host-5 using a EVPN multihoming active-active interface
- Switch device: SW1, connects Host-3 to PE1 and PE2

- Host routing is enabled on PE1 and PE2.
- IRB interfaces are configured as anycast.
- Host-1 and Host-2 connect to PE1 and PE2 using a EVPN single-homing interfaces.
- Host-3 connect to PE1 and PE2 using a EVPN multihoming interfaces.
- IOS XR Software performs the Designated Forwarder (DF) election for the shared Ethernet Segment Identifier (ESI) and elects IF2 as DF and IF5 as Non-Designated Forwarder (NDF).
- Interface IF5 is set to blocked state which blocks both BUM and unicast traffic.

Workflow

Figure 3: EVPN IRB single-active multihoming network topology



Let's consider these scenarios:

- Where Host-3 wants to communicate with Host-5, which are in different subnet.
  1. Host-3 sends an ARP request to its IRB gateway, configured with the BVI IP address 10.0.0.5.
  2. SW1 learns Host-3 MAC and IP addresses as it forwards the ARP response to PE1, which is the DF. The packet sent to PE2 is dropped since IF5 is in a blocked state as a NDF.
  3. PE1 learns Host-3's MAC and IP addresses from the ARP packet and replicates this information across all output interfaces associated with the BVI interface.
  4. The BVI interface on PE1 sends an ARP response to Host-3 using its BVI IP address 10.0.0.5 and MAC address 0.0.5.

   a. SW1 updates its MAC address table with the BVI MAC address of PE1 as it forwards the ARP response to Host-3.

5. PE1 advertises Host-3 host route through EVPN using route type-2. Remote PEs, including PE2 and PE5, learn about Host-3 and install the route in their hardware tables.
6. Since Host-5 is directly connected to PE5, it receives an ARP request and responds with a unicast ARP response.
7. The ARP process ensures that PE5 learns Host-5 IP address 20.0.0.1/32, enabling communication between Host-5 and Host-3.

- Where Host-5 sends a packet to Host-3. If Host-3 hasn't communicated yet, PE5 might not have Host-3 specific route.
  - If PE5 directs traffic to PE2, a Generalized Learning (G-LEAN) adjacency process occurs, and traffic is dropped until it is resolved.
  - If PE2 receives, it performs these steps:
    1. PE2 floods an ARP request within the bridge domain to resolve Host-3 MAC address. However, as PE2 is the NDF, direct forwarding to Host-3 is not possible due to the blocked state of IF5.
    2. A copy of the ARP request is sent to PE1 through the L2 stretch.
    3. PE1 forwards the ARP request to Host-3. Once Host-3 responds to the ARP request, PE1 learns Host-3 MAC address.
    4. After receiving the ARP response, PE1 updates the route for Host-3 and advertises it as an EVPN route type-2 across the network.
    5. This allows packets from Host-5 to reach Host-3 efficiently once the address resolution is complete.

Configure EVPN Single-Active Multihoming

Perform the following steps on PE1 and PE2 to configure EVPN single-active multihoming feature:

Procedure

Step 1 Perform the following steps to configure EVPN IRB with host routing:

a) Configure bridge domain and enable IRB by associating a BVI and interface bundle to create a L2 broadcast domain and provide L3 routing capabilities.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group 6005
Router(config-l2vpn-bg)# bridge-domain 6005
Router(config-l2vpn-bg-bd)# routed interface BVI50
Router(config-l2vpn-bg-bd-bvi)# exit
Router(config-l2vpn-bg-bd-bvi)# interface Bundle-Ether2.1
```

b) Configure EVPN EVI to associate a bridge domain with an Ethernet VPN instance.

Example:

For MPLS core:

```
Router(config-l2vpn-bg-bd-ac)# evi 6005
Router(config-l2vpnbg-bd-evi)# commit
Router(config-l2vpnbg-bd-evi)# exit
```

Example:

For SRv6 core:

```
Router(config-l2vpn-bg-bd-ac)# evi 6005 segment-routing srv6
Router(config-l2vpnbg-bd-evi)# commit
Router(config-l2vpnbg-bd-evi)# exit
```

c) Configure BVI to provide a Layer 3 routing interface for a Layer 2 bridge domain.

Example:

```
Router(config)# interface BVI50
Router(config-if)# host-routing
Router(config-if)# vrf 30
Router(config-if)# ipv4 address 10.0.0.5 255.0.0.0
Router(config-if)# local-proxy-arp
Router(config-if)# mac-address 1.1.1
Router(config-if)# commit
```

Step 2 Configure EVPN ethernet segment to enable the EVPN single-active multihoming on a bundle interface.

Example:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether1
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 40.00.00.00.00.00.00.00.01
Router(config-evpn-ac-es)# load-balancing-mode single-active
Router(config-evpn-ac-es)# bgp route-target 4000.0000.0001
Router(config-evpn-ac-es)# commit
```

Step 3 Configure EVPN service instance (EVI) parameters to set up the BGP-related parameters like Route Distinguisher (RD) and Route Targets (RTs) for the EVPN instance.

Example:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 6005
Router(config-evpn-evi)# bgp
Router(config-evpn-evi-bgp)# rd 200:50
Router(config-evpn-evi-bgp)# route-target import 100:6005
Router(config-evpn-evi-bgp)# route-target export 100:6005
Router(config-evpn-evi-bgp)# commit
```

Step 4 Configure Layer 2 interface to enable the interface for Layer 2 transport, set its encapsulation, and configure tag rewrite rules.

Example:

```
Router# configure
Router(config)# interface bundle-ether2.1 l2transport
Router(config-subif-l2)# no shutdown
Router(config-subif-l2)# encapsulation dot1q 1
Router(config-subif-l2)# rewrite ingress tag pop 1 symmetric
Router(config-subif-l2)#commit
Router(config-subif-l2)#exit
```

Step 5 Perform the following steps to configure a bridge domain:

a) Configure bridge domain on L2VPN to group multiple L2 interfaces into a single broadcast domain.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group 6005
Router(config-l2vpn-bg)# bridge-domain 6005
Router(config-l2vpn-bg-bd)# interface Bundle-Ether2.1
```

b) Configure an EVPN EVI to associate the bridge domain with an Ethernet VPN instance.

Example:

For MPLS core:

```
Router(config-l2vpn-bg-bd-ac)# evi 6005
Router(config-l2vpnbg-bd-evi)# commit
Router(config-l2vpnbg-bd-evi)# exit
```

Example:

For SRv6 core:

```
Router(config-l2vpn-bg-bd-ac)# evi 6005 segment-routing srv6
Router(config-l2vpnbg-bd-evi)# commit
Router(config-l2vpnbg-bd-evi)# exit
```

Step 6 Configure VRF to set up the virtual routing and forwarding instance and define its import and export route targets for VPN routing.

Example:

```
Router# configure
Router(config)# vrf 30
Router(config-vrf)# address-family ipv4 unicast
Router(config-l2vpn-vrf-af)# import route-target 100:6005
Router(config-l2vpn-vrf-af)# export route-target 100:6005
```

Step 7 For SRv6 core, configure SRv6 specific address family to enable per VRF SID allocation.

Example:

```
Router(config-l2vpn-vrf-af)# segment-routing srv6
Router(config-l2vpn-vrf-af-srv6)# alloc mode per-vrf
Router(config-l2vpn-vrf-af-srv6)# exit
Router(config-l2vpn-vrf-af)# commit
```

EVPN IRB Support

EVPN IRB supports the following scenarios:

- In single-homing scenario, only physical, VLAN, .1q, .1ad, or QinQ access methods are supported.
- In dual-homing scenario, only two PE gateways in a redundancy group are supported.
- Both IPv4 and IPv6 are supported.

Distributed Anycast Gateway

EVPN IRB for the given subnet is configured on all the EVPN PEs that are hosted on this subnet. To facilitate optimal routing while supporting transparent virtual machine mobility, hosts are configured with a single default gateway address for their local subnet. That single (anycast) gateway address is configured with a single (anycast) MAC address on all EVPN PE nodes locally supporting that subnet. This process is repeated for each locally defined subnet requires Anycast Gateway support.

The host-to-host Layer 3 traffic, similar to Layer 3 VPN PE-PE forwarding, is routed on the source EVPN PE to the destination EVPN PE next-hop over an IP or MPLS tunnel, where it is routed again to the directly connected host. Such forwarding is also known as Symmetric IRB because the Layer 3 flows are routed at both the source and destination EVPN PEs.

The following solutions are part of the Distributed Anycast Gateway feature:

EVPN IRB with Active-Active Multi-Homing with Subnet Stretch or Host-Routing across the Fabric

For a bridge domain or subnet that is stretched across remote EVPN PEs, both /32 host routes and MAC routes are distributed in a EVPN overlay control plane to enable Layer 2 and Layer 3 traffic to the end points in a stretched subnet.

This type of multi-homing has the following characteristics:

- Layer 2 or Layer 3 ECMP for the fabric for dual-homed hosts based on Route Type 1 and Route Type 2
- Layer 3 unipath over the Fabric for single-homed hosts based on Route Type 2
- Layer 2 subnet stretch over the fabric
- Layer 2 stretch within redundancy group of leafs with orphan ports

MAC and IP Unicast Control Plane

This use case has following types:

Prefix Routing or No Subnet Stretch

IP reachability across the fabric is established using subnet prefix routes that are advertised using EVPN Route Type 5 with the VPN label and VRF RTs. Host ARP and MAC sync are established across multi-homing

EVPN PEs using MAC+IP Route Type 2 based on a shared ESI to enable local switching through both the multi-homing EVPN PEs.

Host Routing or Stretched Subnet

When a host is discovered through ARP, the MAC and IP Route Type 2 is advertised with both MAC VRF and IP VRF router targets, and with VPN labels for both MAC-VRF and IP-VRF. Particularly, the VRF route targets and Layer 3 VPN label are associated with Route Type 2 to achieve PE-PE IP routing identical to traditional L3VPNs. A remote EVPN PE installs IP/32 entries directly in Layer 3 VRF table through the advertising EVPN PE next-hop with the Layer 3 VPN label encapsulation, much like a Layer 3 VPN imposition PE. This approach avoids the need to install separate adjacency rewrites for each remote host in a stretched subnet. Instead, it inherits a key Layer 3 VPN scale benefit of being able to share a common forwarding rewrite or load-balance resource across all IP host entries reachable through a set of EVPN PEs.

ARP and MAC sync

For hosts that are connected through LAG to more that one EVPN PE, the local host ARP and MAC entries are learnt in data plane on either or both of the multihoming EVPN PEs. Local ARP and MAC entries are synced across the two multihoming EVPN PEs using MAC and IP Route Type 2 based on a shared ESI to enable local switching through both the multihoming EVPN PEs. Essentially, a MAC and IP Route Type 2 that is received with a local ESI causes the installation of a synced MAC entry that points to the local AC port, and a synced ARP entry that is installed on the local BVI interface.

Note Only one Ethernet Flow Point (EFP) is supported per non-Zero ESI per bridge domain or EVI. This is a limitation of EVPN.

MAC and IP Route Re-origination

MAC and IP Route Type 2 received with a local ESI, which is used to sync MAC and ARP entries, is also re-originated from the router that installs a SYNC entry, if the host is not locally learnt and advertised based on local learning. This route re-origination is required to establish overlay IP ECMP paths on remote EVPN PEs, and to minimize traffic hit on local AC link failures, that can result in MAC and IP route withdraw in the overlay.

Intra-subnet Unicast Data Plane

The Layer 2 traffic is bridged on the source EVPN PE using ECMP paths to remote EVPN PEs, established through MAC+IP RT2, for every ES and for every EVI, ES and EAD Route Type 2 routes that are advertised from the local EVPN PEs.

Inter-subnet Unicast Data Plane

Inter-subnet traffic is routed on the source EVPN PEs through overlay ECMP to the destination EVPN PEs next-hops. Data packet are encapsulated with the VPN label advertised from the EVPN PE and tunnel label for the BGP next-hop towards the spine. It is then routed again on the destination EVPN PE using a local ARP adjacency towards the host. IP ECMP on the remote EVPN PEs is established through local and re-originated routes advertised from the local EVPN PEs.

Centralized Anycast Gateway for EVPN IRB

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Centralized Anycast Gateway for EVPN IRB	Release 7.5.1	This feature enables configuring the IRB interface in a central location called the Centralized Anycast Gateway. Such a centralized, single-point makes it easier to configure and maintain, reducing the overall operational cost. It also makes it easy to configure the IRB because it's configured only on the centralized gateway of edge L2 devices. This feature is supported only on Single-active mode and this feature introduces **virtual access-evi** command.

EVPN Integrated Routing and Bridging (IRB) allows hosts across the overlay to communicate with each other across the subnets in the VPN. Distributed anycast gateway allows you to configure EVPN IRB for the given subnet on all the access side EVPN PEs that are hosted on the subnet. The distributed gateway provides inner and intra subnet optimal forwarding.

However, the distributed anycast gateway doesn’t provide a centralized point to perform services such as QOS and ACL. Moreover, distributed gateway brings the complexity of setting up and maintaining IRB L3 domains across leaf or edge nodes. Distributed anycast is a bit complex for mobile networks. Customers are looking at connecting and aggregating mobile antennas using cell site routers to core backbone networks in a simple way especially when access networks aren't complex or multilayered Cos fabrics. There’s a need for a single gateway device to carry the load at a lesser cost and that is easy to configure.

With this feature, you can add a centralized gateway to configure IRB in the central location that is known as Centralized Anycast Gateway. The gateway is centralized on a pair of nodes that act as an interface point between the layer-2 and layer-3 routed networks. The Centralized Anycast Gateway uses anycast IRB.

Centralized Anycast GW supports the following services:

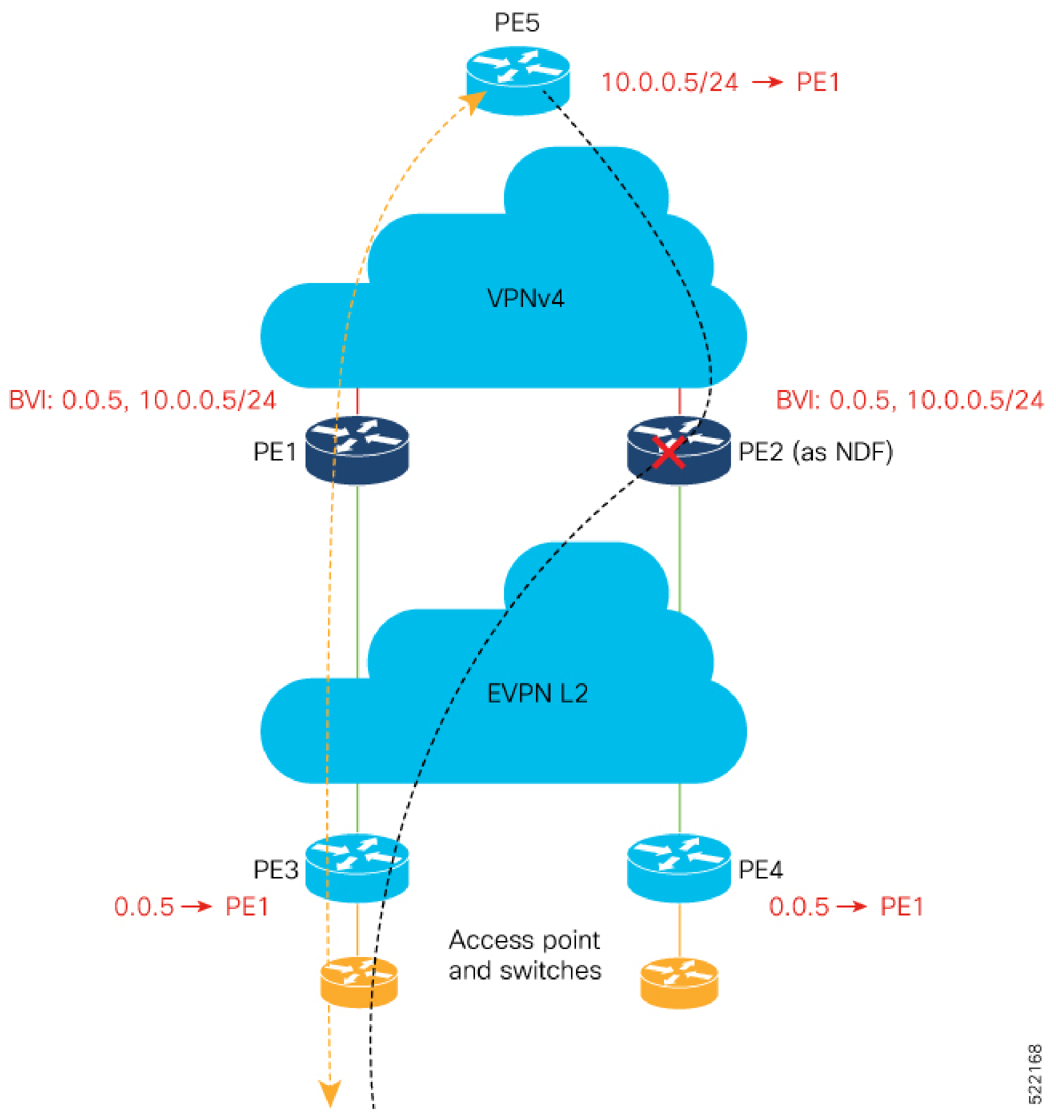
- Unicast Traffic – IPv4/IPv6 on BVI, Global VRF, customer VRF
- Core Isolation
- Cost-in/Cost-out
- DHCP Relay Agent
- MAC Mobility
- Convergence – with Unicast traffic

How Does Centralized Anycast Gateway Work?

EVPN performs DF election between centralized gateways to elect DF and non-DF node at EVI level for load balancing across subnets.

This example shows the centralized gateway without locally attached interface running in a single-active multi-homing mode.

Figure 4: Topology



In this topology, PE1 and PE2 are the centralized gateways where IRB interfaces are configured. Centralized gateways perform EVPN DF election based on EVI/ESI. PE1 is elected as DF whereas PE2 is elected as non-DF.

The following describes the centralized anycast gateway advertisement:

- PE1 is DF and PE2 is non-DF.

- PE1 advertises its EVI to remote PE (PE3 and PE4) where it gets installed as remote entry in the MAC forwarding database. PE2 does not install PE1 as remote MAC.
- PE1 and PE2 advertise their route to the remote PEs (PE3 and PE4).
- PE2 blocks all traffic from EVI to BVI interface.
- MAC-IP addresses of hosts are learned only at the IRB termination point, which is the centralized gateway. PE1 and PE2 advertise host MAC-IP to synchronize with the access EVI.

The following describes how the inter subnet communication happens:

- Host attached to PE3 sends an ARP request to EVPN L2 fabric.
- PE3 learns the host MAC as local and advertises its reachability using EVPN Route-Type2 to remote PEs, including gateways.
- The ARP request reaches PE1 through dataplane and gets blocked on PE2.
- PE1 learns the ARP request locally as the first-hop router.
- PE1 synchronizes ARP entry with PE2 to complete the adjacency.
- PE1 and PE2 receive corresponding host MAC route from BGP and installs it as remote MAC.
- PE1 responds with appropriate ARP reply to source host through PE3.
- PE3 sends unicast traffic to PE1.

The following describes how the intra subnet communication with a host connected to PE4 happens:

- EVPN layer-2 bridging capability where MAC reachability is learned over EVPN Route-Type2.
- Host H3 (attached to PE3) sends direct ARP request to the host IP connected to PE4.
- The PE4 host responds with an ARP reply using the unicast MAC address.

Restrictions

- Physical and bundle AC interfaces are not supported on centralized gateway. Configuration is not prevented but a warning syslog is generated and traffic from locally attached ACs is not supported.
- Only single-active multihoming mode is supported.
- Locally attached interface on gateway is supported only when the access-EVI is configured as Virtual Interface (VES) and EVPN non-DF blocking scheme is used.
- IRB interfaces on peering PE must be configured with anycast gateway.

Configuration

Perform this task to configure centralized gateway. You must have the same configuration on PE1 and PE2:

```
/* EVPN configuration */

Router(config)# evpn
Router(config-evpn)# virtual access-evi
Router(config-evpn-ac-evi)# ethernet-segment
Router(config-evpn-ac-evi-es)# identifier type 0 00.00.00.34.34.34.34.34.34
Router(config-evpn-ac-evi-es)# exit

Router(config-evpn-ac-evi)# core-isolation-group 1
Router(config-evpn-ac-evi)# exit
Router(config-evpn)# exit

/* L2VPN configuration for centralized gateway */

Router(config)# l2vpn
Router(config-l2vpn)#bridge group evpn_access_evi_8
Router(config-l2vpn-bg)#bridge-domain bd_8001
Router(config-l2vpn-bg-bd)#access-evi 8001
Router(config-l2vpn-bg-bd)#routed interface BVI8001
Router(config-l2vpn-bg-bd-bvi)#exit
Router(config-l2vpn-bg-bd)#exit
Router(config-l2vpn-bg)#exit
Router(config-l2vpn)#exit

/* BVI configuration */

Router(config)#Interface BVI8001
Router(config-if)#ipv4 address 10.1.1.1/24
Router(config-if)#mac-address 00aa.8001.00aa
Router(config-if)#commit
```

Show running configuration for centralized gateway:

```
interface BVI8001
 ipv4 address 10.1.1.1 255.255.255.0
 mac-address aa.8001.aa
!
evpn
 virtual access-evi
  ethernet-segment
   identifier type 0 00.00.00.34.34.34.34.34.34
  !
  core-isolation-group 1
 !
 !
l2vpn
 bridge group evpn_access_evi_8
  bridge-domain bd_8001
   access-evi 8001
   routed interface BVI8001
   !
  !
 !
!
```

Perform this task to configure PE2 and PE3:

Note Regular L2 EVPN configuration on PE2 and PE3 and BVI configurations aren't required.

```
Router#configure
Router(config)#evpn
Router(config-evpn)#evi 8001
Router(config-evpn-instance)#advertise-mac
Router(config)#interface bundle-ether80.1 l2transport
Router(config-subif)#encapsulation dot1q 0
Router(config-subif)#rewrite ingress tag pop 1 symmetric
Router(config-subif)#exit

Router(config)#l2vpn
Router(config-l2vpn)#bridge group evpn_access_evi_8
Router(config-l2vpn-bg)#bridge-domain bd_8001
Router(config-l2vpn-bg-bd)#interface bundle-ether 80.1
Router(config-l2vpn-bg-bd-ac)#evi 8001
Router(config-l2vpn-bg-bd-evi)#
```

Verification

Verify the configuration on PE1 and PE2 using the following show commands:

```
Router-PE1# show evpn ethernet-segment carving detail
Ethernet Segment Id      Interface                          Nexthops
------------------------ ---------------------------------- --------------------
0000.0000.3434.3434.3434 Access-EVI:all                     192.168.30.1
                                                            192.168.40.1
Main port         :
     Interface name : Access-EVI/all
Topology          :
Configured     : Single-active (AApS) (default)
  Service Carving   : Auto-selection
Service Carving Results:
     Forwarders     : 200
     Elected        : 100
Not Elected     : 100
```

ARP ND sync for DF and non-DF:

For DF, the state should be `dynamic` and for non-DF, it shows `EVPN-SYNC`.

```
/*For DF:*/
Router-PE1# show arp bvi 9002
Address         Age        Hardware Addr   State      Type  Interface
90.1.2.1        -          0011.0090.1111  Interface  ARPA  BVI9002
90.1.2.34       00:00:46   0012.9001.0002  Dynamic    ARPA  BVI9002

/* For NON-DF*/

Router-PE2# show arp bvi 9002
Address         Age        Hardware Addr   State      Type  Interface
90.1.2.1        -          0011.0090.1111  Interface  ARPA  BVI9002
90.1.2.34       -          0012.9001.0002  EVPN_SYNC  ARPA  BVI9002
```

BVI-Coupled Mode

When ACs go down, the BVI also goes down. However, with this mode enabled, the state of the BVI remains Up even though the ACs go down. Hence, the BVI state becomes EVPN-aware.

BVI tracks the Up or Down state of ACs and PWs in a bridge. When the EVPN port is available, there may be an L2 redirect path over EVI to carry the traffic between L3 to L2. However, this depends on the remote or peer EVI-EAD routes received.

Under certain conditions, you can reduce the churns of BVI state adjacency by keeping the BVI state Up. BVI state drives the state of EVPN_SYNC adjacencies being pushed to forwarding entries, thereby reducing the churns further. Keeping the BVI state Up, the router creates adjacencies in the forwarding table, which indicates that a local adjacency is invalid when an interface is down.

Configure BVI-Coupled Mode

Perform this task to configure BVI-coupled mode.

```
evpn
 evi 101
  bgp
   route-target import 60000:101
   route-target export 60000:101
  !
  bvi-coupled-mode

l2vpn
 bridge group BG-1
  bridge-domain BD-1
   interface Bundle-Ether100.101
   !
   routed interface BVI101
   !
   evi 101
```

Verification

Verify that the BVI-coupled mode is enabled.

```
Router# show evpn evi detail

VPN-ID     Encap      Bridge Domain                Type
---------- ---------- ---------------------------- -------------------
101        MPLS       BD-1                         EVPN
   Stitching: Regular
   Unicast Label  : 35048
   Multicast Label: 33000
   Reroute Label: 0
   Flow Label: N
   Control-Word: Enabled
   E-Tree: Root
   Forward-class: 0
   Advertise MACs: Yes
   Advertise BVI MACs: No
   Aliasing: Enabled
   UUF: Enabled
   Re-origination: Enabled
   Multicast:
     Source connected   : No
     IGMP-Snooping Proxy: No
     MLD-Snooping Proxy : No
   BGP Implicit Import: Enabled
   VRF Name: cust1
   Preferred Nexthop Mode: Off
BVI Coupled Mode: Yes --------------------------> enabled
   BVI Subnet Withheld: ipv4 No, ipv6 No
   RD Config: none
   RD Auto  : (auto) 201.201.201.1:101
   RT Auto  : 60000:101
   Route Targets in Use           Type
```

```
----------------------------- -------------
60000:101                     Import
60000:101                     Export
---------------------- --------------------
```

VM Mobility Support

VM mobility is the ability of virtual machines to migrate between one server and another while retaining their existing MAC and IP addresses.

The following are the two key components in EVPN Route Type 2 that enable VM Mobility:

- Host MAC advertisement component that is imported into local bridge MAC table, and Layer 2 bridged traffic across the network overlay.
- Host IP advertisement component that is imported into the IP routing table in a symmetric IRB design, enables routed traffic across the network overlay.

The above-mentioned components are advertised together in a single MAC + IP host route advertisement. An additional MAC-only route could also be advertised.

The following behaviors of VM are supported. The VM can:

- retain existing MAC and acquire a new IP address
- retain existing IP address and acquire a new MAC
- retain both existing MAC and IP address

Limitation for VM Mobility Support

- You cannot perform VM mobility move with new IP to MAC.

Configure EVPN IRB

Perform these steps to configure a bridge domain. You can perform this task over an MPLS core or an SRv6 core.

Before you begin

If you are deploying over an SRv6 core, make sure you configure SRv6 underlay on the participating routers.

Procedure

Step 1 Configure LACP (Link Aggregation Control Protocol) to set the system MAC for the bundle interface.

Example:

```
Router# configure
Router(config)# interface Bundle-Ether 3
Router(config-if)# lacp system mac 1.1.1
Router(config-if)# exit
```

Step 2 Configure EVPN L3VRF per DC tenant to define the VRF instance and its route targets for IPv4 unicast.

Example:

```
Router# configure
Router(config)# vrf irb1
Router(config-vrf)# address-family ipv4 unicast
Router(config-vrf-af)# import route-target 1000:1
Router(config-vrf-af)# export route-target 1000:1
Router(config-vrf-af)# exit
```

Step 3 Configure Layer 2 Attachment Circuit (AC) from Multichassis (MC) bundle interface, and Bridge-group Virtual Interface (BVI) per bridge domain to enable host routing capabilities within the VRF.

Example:

When a VM migrates from one subnet to another (subnet stretching), apply the following IRB configuration to both the EVPN PEs.

```
Router# configure
Router(config)# interface bvi 1001
Router(config-if)# host-routing
Router(config-if)# vrf irb1
Router(config-if)# ipv4 address 10.10.0.4 255.255.255.0
Router(config-if)# ipv4 address 172.16.0.1 secondary
Router(config-if)# mac-address 00aa.1001.00aa
```

Step 4 Configure EVPN Layer 2 bridging service to define a bridge group and bridge domain for Layer 2 gateway or bridging scenarios.

Example:

The following configuration is performed in Layer 2 gateway or bridging scenario.

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group 1
Router(config-l2vpn-bg)# bridge-domain 1-1
Router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/0/0/1.1
```

Step 5 Configure an EVPN EVI for EVPN Layer 2 bridging service to associate the bridge domain with an Ethernet VPN instance.

Example:

For MPLS core:

```
Router(config-l2vpn-bg-bd-ac)# evi 1
Router(config-l2vpn-bg-bd-ac-evi)# commit
```

Example:

For SRv6 core:

```
Router(config-l2vpn-bg-bd-ac)# evi 1 segment-routing srv6
Router(config-l2vpn-bg-bd-ac-evi)# commit
```

Step 6 Configure BGP to enable routing for the VRF.

Example:

For MPLS core:

```
Router# configure
Router(config)# router bgp 3107
Router(config-bgp)# vrf irb1
Router(config-bgp-vrf)# rd auto
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# redistribute connected
Router(config-bgp-vrf-af)# redistribute static
Router(config-bgp-vrf-af)# exit
```

Example:

For SRv6 core:

```
Router# configure
Router(config)# vrf 30
Router(config-vrf)# address-family ipv4 unicast
Router(config-l2vpn-vrf-af)# import route-target 100:6005
Router(config-l2vpn-vrf-af)# export route-target 100:6005
Router(config-l2vpn-vrf-af)# segment-routing srv6
Router(config-l2vpn-vrf-af-srv6)# alloc mode per-vrf
Router(config-l2vpn-vrf-af-srv6)# exit
Router(config-l2vpn-vrf-af)# commit
```

Step 7 Configure an EVPN EVI for main bundle ethernet segment parameters in EVPN to define the EVPN instance.

Example:

For MPLS core:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 2001
```

Example:

For SRv6 core:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 2001 segment-routing srv6
```

Step 8 Configure main bundle ethernet segment parameters in EVPN to set BGP route targets.

Example:

```
Router(config-evpn-evi)# bgp
Router(config-evpn-evi-bgp)# route-target import 1000:1
Router(config-evpn-evi-bgp)# route-target export 1000:1
Router(config-evpn-evi-bgp)# exit
Router(config-evpn-evi)# advertise-mac
Router(config-evpn-evi)# unknown-unicast-suppression
```

Step 9 Configure Layer 2 VPN to define a bridge group and bridge domain.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group irb
Router(config-l2vpn-bg)# bridge-domain irb1
Router(config-l2vpn-bg-bd)# interface bundle-Ether3.1001
Router(config-l2vpn-bg-bd-ac)# routed interface BVI100
Router(config-l2vpn-bg-bd-bvi)# split-horizon group core
```

Step 10 Configure an EVPN EVI for Layer 2 VPN to define the EVPN instance for the L2VPN service.

Example:

For MPLS core:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 1001
```

Example:

For SRv6 core:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 2001 segment-routing srv6
```

Step 11 Run the **show arp vrf** command to verify the Address Resolution Protocol (ARP) protocol entries and synced entries in multi-homing scenarios.

Example:

```
Router# show arp vrf evpn1
-----------------------------------------------------------------
0/1/CPU0
-----------------------------------------------------------------
Address Age Hardware Addr State Type Interface
10.1.1.1 - 0010.0001.0001 Interface ARPA BVI1
10.1.1.11 02:23:46 1000.0001.0001 Dynamic ARPA BVI1
10.1.1.93 - 0000.f65a.357c EVPN_SYNC ARPA BVI1
10.1.2.1 - 0011.0112.0001 Interface ARPA BVI2
10.1.2.91 02:24:14 0000.f65a.3570 Dynamic ARPA BVI2
10.1.2.93 02:21:52 0000.f65a.357d Dynamic ARPA BVI2
---------------------------------------------------------------
0/0/CPU0
---------------------------------------------------------------
Address Age Hardware Addr State Type Interface
10.1.1.1 - 0010.0001.0001 Interface ARPA BVI1
10.1.1.11 02:23:46 1000.0001.0001 Dynamic ARPA BVI1
10.1.1.93 - 0000.f65a.357c EVPN_SYNC ARPA BVI1
10.1.2.1 - 0011.0112.0001 Interface ARPA BVI2
10.1.2.91 02:24:14 0000.f65a.3570 Dynamic ARPA BVI2
10.1.2.93 02:21:52 0000.f65a.357d Dynamic ARPA BVI2
```

Step 12 Run the **show adjacency ipv4** command to verify the adjacency entries, particularly newly added information for synced IPv4 and IP ARP entries.

Example:

```
Router# show adjacency ipv4 BVI 1 internal detail location 0/0/CPU0
BVI1, 10.1.1.93 (ipv4)
Version: 1169, references: 2, transient lock: 0
Encapsulation information (14 bytes) 0000f65a357c0000f65a357c0800 MTU: 1500
Adjacency pointer is: 0x770a9278
Platform adjacency pointer is: 0x7d7bc380
Last updated: Feb 28 15:58:21.998
Adjacency producer: arp (prod_id: 10)
Flags: incomplete adj,
Additional Adjacency Information (4 bytes long),
Upto first 4 bytes (in hex): 01000000
Netio idb pointer not cached Cached interface type: 78
Adjacency references:
bfd_agent (JID 150, PID 3637), 0 reference
l2fib_mgr (JID 185, PID 4003), 0 reference
fib_mgr (JID 294, PID 3605), 1 reference
aib (JID 314, PID 3590), 1 reference
BVI1, 10.1.1.11 (ipv4) Version: 1493,
references: 3, transient lock: 0
Encapsulation information (14 bytes) 1000000100010010000100010800
MTU: 1500
Adjacency pointer is: 0x770ab778
Platform adjacency pointer is: 0x7d7bcb10
Last updated: Mar 2 17:22:00.544
Adjacency producer: arp (prod_id: 10)
Flags: incomplete adj,
Netio idb pointer not cached Cached interface type: 78
Adjacency references:
bfd_agent (JID 150, PID 3637), 0 reference
l2fib_mgr (JID 185, PID 4003), 1 reference
fib_mgr (JID 294, PID 3605), 1 reference
aib (JID 314, PID 3590), 1 reference
```

Step 13 Run the **show l2vpn mac-learning** command to verify the entries to obtain details learned in L2FIB line cards, including link-local addresses updated and distributed in multi-homing active-active scenarios.

Example:

```
Router# show l2vpn mac-learning mac-ipv4 all location 0/0/cPU0
Topo ID Producer Next Hop(s) Mac Address IP Address
6 0/0/CPU0 BV1 1000.0001.0001 10.1.1.11
7 0/0/CPU0 BV2 0000.f65a.3570 10.1.2.91
7 0/0/CPU0 BV2 0000.f65a.357d 10.1.2.93
```

Step 14 Run the **show l2route evpn mac-ip** command to verify the sequence ID for VM mobility.

Example:

```
Router# show l2route evpn mac-ip all detail
Flags: (Stt)=Static; (L)=Local; (R)=Remote; (F)=Flood;
(N)=No Redistribution; (Rtr)=Router MAC; (B)=Best Route;
(P)=Probe; (S)=Peer Sync; (F)=Flush;
(D)=Duplicate MAC; (Z)=Frozen MAC;
Topo ID Mac Address IP Address Prod Next Hop(s) Seq No Flags
Opaque Data Type Opaque Data Len Opaque Data Value
------- ----------- ---------- ---- ---------- ------ -----
---------------- -------------- -----------------
33 0022.6730.0001 10.130.0.2 L2VPN Bundle-Ether6.1300 0 SB 0 12
0x06000000 0x22000080 0x00000000
Last Update: Sun Apr 30 15:00:01.911 PDT
33 0022.6730.0002 10.130.0.3 LOCAL Bundle-Ether6.1300 0 B N/A
N/A N/A
```

Step 15 Run the **show l2vpn mac-learning** command to verify the entries to obtain details learned in L2FIB RP, which are aggregated entries from line cards, to observe MAC move states and updates to L2RIB.

Example:

```
Router# show l2vpn mac-learning mac-ipv4 all location 0/RSP0/CPU0
Topo ID Producer Next Hop(s) Mac Address IP Address
------- -------- ----------- -------------- ----------
6 0/0/CPU0 BV1 1000.0001.0001 10.1.1.11
7 0/0/CPU0 BV2 0000.f65a.3570 10.1.2.91
7 0/0/CPU0 BV2 0000.f65a.357d 10.1.2.93
```

Step 16 Run the **show l2route evpn** command to verify the entries in L2RIB that are updated by RP L2FIB.

Example:

Note the following when you verify the entries:

- The entries with producer as L2VPN and NH as remote IP are learnt from the remote peer gateways, which are learnt from BGP, updated to EVPN, and then updated to L2RIB. So these entries are not from local IP-MAC learning.
- The entries with producer as L2VPN and NH as local bundle interfaces are synced entries from MH-AA peer gateway.
- The entries with producer as LOCAL and NH as local bundle interfaces are dynamically learnt local entries.

```
Router# show l2route evpn mac-ip evi 6
Topo ID Mac Address IP Address Prod Next Hop(s)
-------- -------------- --------------- ------ -------------
6 0000.f65a.3569 10.1.1.101 L2VPN 172.16.0.2/24014/ME
6 0000.f65a.3575 10.1.1.97 L2VPN 172.16.0.7/24025/ME
6 0000.f65a.3575 10:1:1::97 L2VPN 172.16.0.7/24025/ME
6 0000.f65a.3575 fe80::200:f6ff:fe5a:3575 L2VPN 172.16.0.7/24025/ME
6 0000.f65a.357c 10.1.1.93 L2VPN Bundle-Ether1.11
6 0000.f65a.357c 10:1:1::93 L2VPN Bundle-Ether1.11
6 0000.f65a.357c fe80::200:f6ff:fe5a:357c LOCAL Bundle-Ether1.11
6 0010.0001.0012 10.1.1.12 L2VPN 172.16.0.7/24025/ME
6 1000.0001.0001 10.1.1.11 LOCAL Bundle-Ether1.11
6 90e2.ba8e.c0c9 10.1.1.102 L2VPN 172.16.0.2/24014/ME
```

Step 17 Run the **show evpn evi** command to obtain details of EVPN.

Example:

For MPLS core:

```
Router# show evpn evi vpn-id 1 mac ipv4 10.1.1.93 detail
EVI MAC address IP address Nexthop Label
---- --------------- ---------- ---------- -----
1 0000.f65a.357c 10.1.1.93 172.16.0.2 24014
Ethernet Tag : 0
Multi-paths Resolved : True
Static : No
Local Ethernet Segment : N/A
Remote Ethernet Segment : 0100.6cbc.a77c.c180.0000
Local Sequence Number : N/A
Remote Sequence Number : 0
Local Encapsulation : N/A
Remote Encapsulation : MPLS
```

Example:

For SRv6 core:

```
Router# show evpn evi  vpn-id 100 mac ipv4 171.1.1.2 det
VPN-ID     Encap      MAC address    IP address                               Nexthop
                    Label    SID
---------- ---------- -------------- ----------------------------------------
---------------------------------------- -------- ----------------------------------------
100        SRv6       0011.0100.0001 171.1.1.2                                fccc:cc00:1101::1
                    IMP-NULL fccc:cc03:1031:e000::
100        SRv6       0011.0100.0001 171.1.1.2                                fccc:cc00:1102::1
                    IMP-NULL fccc:cc03:1032:e000::
   Ethernet Tag                           : 0
   Multi-paths Resolved                   : True
   Multi-paths Internal label             : None
   Multi-paths Internal ID                : ::ffff:10.0.0.1
   Service Group ID                       : None
   Local Static                           : No
   Remote Static                          : No
   Local Ethernet Segment                 : N/A
   Remote Ethernet Segment               : 0011.1111.1111.1111.1111
   Local Sequence Number                 : N/A
   Remote Sequence Number                : 2
   Local Encapsulation                   : N/A
   Remote Encapsulation                  : SRv6
   Local E-Tree                          : Root
   Remote E-Tree                         : Root
   Remote matching E-Tree RT             : No
   Local AC-ID                           : 0x0
   Remote AC-ID                          : 0x64
   Local ARP/ND Information              : N/A
   Remote ARP/ND Information
       Immutable                         : No
       ND Override                       : No
       ND Router                         : No
```

Step 18 Run the **show bgp l2vpn evpn** command to verify local BGP entries with appropriate second label and second IP VRF route-target.

Example:

For MPLS core:

```
Router# show bgp l2vpn evpn rd 172.16.0.1:1
[2][0][48][0000.f65a.357c][32][10.1.1.93]/136
BGP routing table entry for [2][0][48][0000.f65a.357c][32][10.1.1.93]/136, Route
Distinguisher: 172.16.0.1:1
Versions:
Process bRIB/RIB SendTblVer
Speaker 3772 3772
Local Label: 24013
Last Modified: Feb 28 16:06:37.073 for 2d19h
Paths: (2 available, best #1)
Advertised to peers (in unique update groups):
172.16.0.9
Path #1: Received by speaker 0
Advertised to peers (in unique update groups):
172.16.0.9
Local
0.0.0.0 from 0.0.0.0 (172.16.0.1)
Second Label 24027 >>>> Second label when IRB host-routing
is enabled.
Origin IGP, localpref 100, valid, redistributed, best, group-best, import-candidate,
rib-install
Received Path ID 0, Local Path ID 0, version 3772
Extended community: SoO:172.16.0.2:1 RT:100:100
EVPN ESI: 0100.6cbc.a77c.c180.0000
Path #2: Received by speaker 0
Not advertised to any peer
Local
172.16.0.2 (metric 101) from 172.16.0.9 (172.16.0.2)
Received Label 24014, Second Label 24031
Origin IGP, localpref 100, valid, internal, add-path, import-candidate, imported, rib-install
Received Path ID 0, Local Path ID 2, version 3769
Extended community: SoO:172.16.0.2:1 RT:200:1 RT:700:100 >>> Second RT is IP VRF RT for
remote to import into IP VRF routing table.
Originator: 172.16.0.2, Cluster list: 172.16.0.9
EVPN ESI: 0100.6cbc.a77c.c180.0000
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.2:1
```

Example:

For SRv6 core:

```
Router# sh bgp l2vpn evpn bridge-domain elan-1001 [2][0][48][0022.0222.0001][32][2.0.1.2]/136 detail
BGP routing table entry for [2][0][48][0022.0222.0001][32][2.0.1.2]/136, Route Distinguisher:
11.11.11.11:1001
Versions:
 Process bRIB/RIB SendTblVer
 Speaker 1939     1939
  Flags: 0x00001001+0x00010000+0x00000000
Last Modified: Sep 10 14:30:30.223 for 00:00:19
Paths: (2 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Flags: 0x2000020005060005+0x00, import: 0x080, EVPN: 0x3
  Not advertised to any peer
  Local
    2::1 (metric 10) from 2::1 (22.22.22.22), if-handle 0x00000000
      Received Label 0xf00007, Second Label 0x3 >>> L2 Sid + Received label for L2 MAC unicast
bridging; L3 SID + Second label for EVPN IRB host-routing
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported,
rib-install
      Received Path ID 0, Local Path ID 1, version 1939
      Extended community: SoO:33.33.33.33:1001 EVPN ARP/ND:0x00 0x060e:0000.00bb.93e9 RT:100:1001

      EVPN ESI: 0000.0000.0000.0000.0201
      PSID-Type:L3, SubTLV Count:1, R:0x00,
        SubTLV:
          T:1(Sid information), Sid:fccc:cc00:2:e018::, F:0x00, R2:0x00, Behavior:63, R3:0x00,
SS-TLV Count:1
           SubSubTLV:
            T:1(Sid structure):
            Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
      PSID-Type:L2, SubTLV Count:1, R:0x00,
        SubTLV:
          T:1(Sid information), Sid:fccc:cc00:2:ff00::(Transposed), F:0x00, R2:0x00, Behavior:67,
R3:0x00, SS-TLV Count:1
            SubSubTLV:
          T:1(Sid structure):
          Length [Loc-blk,Loc-node,Func,Arg]:[32,16,32,0], Tpose-len:24, Tpose-offset:56
       Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 22.22.22.22:1001
  Path #2: Received by speaker 0
  Flags: 0x2000020004020005+0x00, import: 0x080, EVPN: 0x3
  Not advertised to any peer
  Local
    3::1 (metric 10) from 3::1 (33.33.33.33), if-handle 0x00000000
      Received Label 0xe00000, Second Label 0x3
      Origin IGP, localpref 100, valid, internal, import-candidate, imported, rib-install
      Received Path ID 0, Local Path ID 0, version 0
      Extended community: SoO:33.33.33.33:1001 EVPN ARP/ND:0x00 0x060e:0000.00bb.93e9 RT:100:1001
      EVPN ESI: 0000.0000.0000.0000.0201
      PSID-Type:L3, SubTLV Count:1, R:0x00,
        SubTLV:
          T:1(Sid information), Sid:fccc:cc00:3:e016::, F:0x00, R2:0x00, Behavior:63, R3:0x00,
SS-TLV Count:1
          SubSubTLV:
            T:1(Sid structure):
            Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
      PSID-Type:L2, SubTLV Count:1, R:0x00,
        SubTLV:
         T:1(Sid information), Sid:fccc:cc00:3::(Transposed), F:0x00, R2:0x00, Behavior:67, R3:0x00,
 SS-TLV Count:1
          SubSubTLV:
            T:1(Sid structure):
            Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:16, Tpose-offset:48
       Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 33.33.33.33:1001
```

Step 19 Run the **show bgp l2vpn evpn** command to verify the remote peer gateway BGP entries with correct label and route-target. Particularly verify the local auto-generated RD on a remote EVPN gateway. EVPN type-2 routes are imported into EVPN. The host routes of IPv4 /32 addresses are imported only into IP VRF route-table in the remote EVPN gateway, but not in the local EVPN gateway where local BVI adjacency is used to overwrite RIB entries.

Example:

Example:

```
Router# show bgp l2vpn evpn rd 172.16.0.7:1
[2][0][48][0000.f65a.357c][32][10.1.1.93]/136
BGP routing table entry for [2][0][48][0000.f65a.357c][32][10.1.1.93]/136, Route
Distinguisher: 172.16.0.7:1
Versions:
Process bRIB/RIB SendTblVer
Speaker 16712 16712
Last Modified: Feb 28 16:06:36.448 for 2d19h
Paths: (2 available, best #1)
Not advertised to any peer
Path #1: Received by speaker 0
Not advertised to any peer
Local
172.16.0.1 from 172.16.0.9 (172.16.0.1)
Received Label 24013, Second Label 24027 >>>> First label for L2 MAC unicast bridging;
second label for EVPN IRB host-routing
Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported,
rib-install
Received Path ID 0, Local Path ID 0, version 16712
Extended community: SoO:172.16.0.2:1 RT:100:1 RT:100:100
Originator: 172.16.0.1, Cluster list: 172.16.0.9
EVPN ESI: 0100.6cbc.a77c.c180.0000
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.1:1
Path #2: Received by speaker 0
Not advertised to any peer
Local
172.16.0.2 from 172.16.0.9 (172.16.0.2)
Received Label 24014, Second Label 24031
Origin IGP, localpref 100, valid, internal, backup, add-path, import-candidate, imported,
rib-install
Received Path ID 0, Local Path ID 1, version 16706
Extended community: SoO:172.16.0.2:1 RT:200:1 RT:700:100
Originator: 172.16.0.2, Cluster list: 172.16.0.9
EVPN ESI: 0100.6cbc.a77c.c180.0000
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.2:1
```

Step 20 Run the **show bgp vpnv4 unicast vrf** and **show bgp vpnv6 unicast vrf** commands to verify the remote peer gateway with host routes of IPv4 /32 addresses imported into the IP VRF routing table.

Example:

```
Router# show bgp vpnv4 unicast vrf evpn1 10.1.1.93/32
BGP routing table entry for 10.1.1.93/32, Route Distinguisher: 172.16.0.7:11
Versions:
Process bRIB/RIB SendTblVer
Speaker 22202 22202
Last Modified: Feb 28 16:06:36.447 for 2d19h
Paths: (2 available, best #1)
Not advertised to any peer
Path #1: Received by speaker 0
Not advertised to any peer
Local
172.16.0.1 from 172.16.0.9 (172.16.0.1)
Received Label 24027
Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported
```

```
Received Path ID 0, Local Path ID 0, version 22202
Extended community: SoO:172.16.0.2:1 RT:100:1 RT:100:100
Originator: 172.16.0.1, Cluster list: 172.16.0.9
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.1:1 >>>>
The source from
>>>> L2VPN and from
>>>> synced ARP entry.
Path #2: Received by speaker 0
Not advertised to any peer
Local
172.16.0.2 from 172.16.0.9 (172.16.0.2)
Received Label 24031
Origin IGP, localpref 100, valid, internal, backup, add-path, import-candidate, imported
Received Path ID 0, Local Path ID 1, version 22201
Extended community: SoO:172.16.0.2:1 RT:200:1 RT:700:100
Originator: 172.16.0.2, Cluster list: 17.0.0.9
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.2:1 >>>>
The source from
>>>> L2VPN and
>>>> from dynamic
>>>> ARP entry.
```

Example:

```
Router# show bgp vpnv6 unicast vrf evpn1 10:1:1::93/128
BGP routing table entry for 10:1:1::93/128, Route Distinguisher: 172.16.0.7:11
Versions:
Process bRIB/RIB SendTblVer
Speaker 22163 22163
Last Modified: Feb 28 12:09:30.447 for 2d23h
Paths: (2 available, best #1)
Not advertised to any peer
Path #1: Received by speaker 0
Not advertised to any peer
Local
172.16.0.1 from 172.16.0.9 (172.16.0.1)
Received Label 24029
Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported
Received Path ID 0, Local Path ID 0, version 22163
Extended community: SoO:172.16.0.2:1 RT:100:1 RT:100:100
Originator: 172.16.0.1, Cluster list: 172.16.0.9
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.1:1 >>>>
Source from
>>>> L2VPN and from
>>>> synced ARP entry.
Path #2: Received by speaker 0
Not advertised to any peer
Local
172.16.0.2 from 172.16.0.9 (172.16.0.2)
Received Label 24033
Origin IGP, localpref 100, valid, internal, backup, add-path, import-candidate, imported
Received Path ID 0, Local Path ID 1, version 22163
Extended community: SoO:172.16.0.2:1 RT:200:1 RT:700:100
Originator: 172.16.0.2, Cluster list: 172.16.0.9
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.2:1 >>>>
Source from
>>>> L2VPN and from
>>>> dynamic ARP entry.
```

Example:

```
Router# show bgp vpnv6 unicast vrf evpn1 10:1:1::93/128
BGP routing table entry for 10:1:1::93/128, Route Distinguisher: 172.16.0.7:11
Versions:
Process bRIB/RIB SendTblVer
Speaker 22163 22163
```

```
Last Modified: Feb 28 12:09:30.447 for 2d23h
Paths: (2 available, best #1)
Not advertised to any peer
Path #1: Received by speaker 0
Not advertised to any peer
Local
172.16.0.1 from 172.16.0.9 (172.16.0.1)
Received Label 24029
Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported
Received Path ID 0, Local Path ID 0, version 22163
Extended community: SoO:172.16.0.2:1 RT:100:1 RT:100:100
Originator: 172.16.0.1, Cluster list: 172.16.0.9
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.1:1
Path #2: Received by speaker 0
Not advertised to any peer
Local
172.16.0.2 from 172.16.0.9 (172.16.0.2)
Received Label 24033
Origin IGP, localpref 100, valid, internal, backup, add-path, import-candidate, imported
Received Path ID 0, Local Path ID 1, version 22163
Extended community: SoO:172.16.0.2:1 RT:200:1 RT:700:100
Originator: 172.16.0.2, Cluster list: 172.16.0.9
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 172.16.0.2:1
```

Step 21 Run the **show bgp vpnv4 unicast vrf** command to local forwarding with local adjacency which overwrite the RIB entries, and remote peer that use the IP VRF host route entries for IP VPN forwarding.

Example:

For MPLS core:

```
Router# show bgp vpnv4 unicast vrf evpn1 10.1.1.93/32
-- For local routing and forwarding
RP/0/RSP0/CPU0:PE11-R1#show route vrf evpn1 10.1.1.93
Routing entry for 10.1.1.93/32
Known via "bgp 3107", distance 200, metric 0, type internal
Installed Feb 28 15:57:28.154 for 2d20h
Routing Descriptor Blocks
172.16.0.2, from 172.16.0.9 >>> From MH-AA peer.
Nexthop in Vrf: "default", Table: "default", IPv4 Unicast, Table Id: 0xe0000000
Route metric is 0
No advertising protos.
```

Example:

For SRv6 core:

```
Router# show bgp vpnv4 unicast vrf CE100 171.1.1.2/32
BGP routing table entry for 171.1.1.2/32, Route Distinguisher: 100:100
Versions:
  Process           bRIB/RIB   SendTblVer
  Speaker            1145764      1145764
Last Modified: Nov 10 14:32:33.063 for 00:02:39
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    fccc:cc00:1102::1 (metric 20) from fccc:cc00:1301::1 (1.1.1.2)
      Received Label 0xe0180
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported
      Received Path ID 1, Local Path ID 1, version 1145764
      Extended community: SoO:1.1.1.2:100 Color:100 EVPN MAC Mobility:0x00:2 EVPN ARP/ND:0x00
0x060e:0000.0000.0064 RT:100:100 RT:60100:100
      mac: 00:11:01:00:00:01
      ethernet tag: 0
      Originator: 1.1.1.2, Cluster list: 1.1.1.5
      PSID-Type:L3, SubTLV Count:1
       SID Locator: fccc:cc03:1032::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1032::(Transposed), Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.2:100
```

Example:

For MPLS core:

```
Router# show cef vrf evpn1 10.1.1.93 location 0/0/CPU0
10.1.1.93/32, version 0, internal 0x1120001 0x0 (ptr 0x7b40052c) [1], 0x0 (0x7b286010), 0x0
(0x0)
Updated Feb 28 15:58:22.688
local adjacency 10.1.1.93
Prefix Len 32, traffic index 0, Adjacency-prefix, precedence n/a, priority 15
via 10.1.1.93/32, BVI1, 2 dependencies, weight 0, class 0 [flags 0x0]
path-idx 0 NHID 0x0 [0x7f531f88 0x0]
next hop
local adjacency >>> Forwarding with local synced ARP adjacency entries.
For remote routing and forwarding:
```

Example:

For SRv6 core:

```
Router# show cef vrf CE100 171.1.1.2
171.1.1.2/32, version 0, internal 0x1120001 0xf0 (ptr 0xa7000ec8) [1], 0x400 (0xa3be6b98), 0x0 (0x0)

 Updated Nov 10 13:53:41.756
 local adjacency to BVI100

 Prefix Len 32, traffic index 0, Adjacency-prefix, precedence n/a, priority 15
  gateway array (0x8bd53a98) reference count 100, flags 0x800000, source aib (3), 0 backups
                [101 type 3 flags 0x1008401 (0x8bdd0018) ext 0x0 (0x0)]
  LW-LDI[type=3, refc=1, ptr=0xa3be6b98, sh-ldi=0x8bdd0018]
  gateway array update type-time 1 Nov 10 13:53:41.767
 LDI Update time Nov 10 12:33:15.128
 LW-LDI-TS Nov 10 13:53:41.756
   via 171.1.1.2/32, BVI100, 2 dependencies, weight 0, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0xa727e4a0 0x0]
    next hop
    local adjacency

    Load distribution: 0 (refcount 101)

    Hash  OK  Interface                 Address
    0     Y   BVI100                    Shared
```

Example:

For MPLS core:

```
Router# show route vrf evpn1 10.1.1.93
Routing entry for 10.1.1.93/32
Known via "bgp 3107", distance 200, metric 0
Number of pic paths 1 , type internal
Installed Feb 28 16:06:36.431 for 2d20h
Routing Descriptor Blocks
172.16.0.1, from 172.16.0.9
Nexthop in Vrf: "default", Table: "default", IPv4 Unicast, Table Id: 0xe0000000
Route metric is 0
172.16.0.2, from 172.16.0.9, BGP backup path
```

```
Nexthop in Vrf: "default", Table: "default", IPv4 Unicast, Table Id: 0xe0000000
Route metric is 0
No advertising protos.
```

Example:

For SRv6 core:

```
Router# show route vrf CE100 171.1.1.2
Routing entry for 171.1.1.2/32
  Known via "application l2vpn_evpn", distance 2, metric 0
  Installed Nov 10 13:53:41.642 for 00:41:36
  Routing Descriptor Blocks
    directly connected, via BVI100
      Route metric is 0
   No redist to FIB
  No advertising protos
```

Example:

```
Router# show cef vrf evpn1 10.1.1.93 location 0/0/CPU0
10.1.1.93/32, version 86, internal 0x5000001 0x0 (ptr 0x99fac884) [1], 0x0 (0x0), 0x208
(0x96c58494)
Updated Feb 28 16:06:39.285
Prefix Len 32, traffic index 0, precedence n/a, priority 3
via 172.16.0.1/32, 15 dependencies, recursive [flags 0x6000]
path-idx 0 NHID 0x0 [0x97955380 0x0]
recursion-via-/32
next hop VRF - 'default', table - 0xe0000000
next hop 172.16.0.1/32 via 34034/0/21
next hop 100.0.57.5/32 Te0/0/0/3 labels imposed {ImplNull 24011 24027}
next hop 100.0.67.6/32 Te0/0/0/1 labels imposed {ImplNull 24009 24027}
via 172.16.0.2/32, 11 dependencies, recursive, backup [flags 0x6100]
path-idx 1 NHID 0x0 [0x979554a0 0x0]
recursion-via-/32
next hop VRF - 'default', table - 0xe0000000
next hop 172.16.0.2/32 via 34035/0/21
next hop 100.0.57.5/32 Te0/0/0/3 labels imposed {ImplNull 24012 24031}
next hop 100.0.67.6/32 Te0/0/0/1 labels imposed {ImplNull 24010 24031}
```

Example:

For SRv6 core:

```
Router# show bgp l2vpn evpn rd 1.1.1.1:100  [2][0][48][0011.0100.0001][32][171.1.1.2]/136
BGP routing table entry for [2][0][48][0011.0100.0001][32][171.1.1.2]/136, Route Distinguisher:
1.1.1.1:100
Versions:
  Process           bRIB/RIB   SendTblVer
  Speaker             3042441      3042441
    Local Label: 0xe00000
Last Modified: Nov 10 14:32:33.063 for 00:02:30
Paths: (2 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.2
  Path #1: Received by speaker 0
  Advertised to update-groups (with more than one peer):
    0.2
  Local
    0.0.0.0 from 0.0.0.0 (1.1.1.1)
      Second Label 0xe0220
     Origin IGP, localpref 100, valid, redistributed, best, group-best, import-candidate, rib-install

      Received Path ID 0, Local Path ID 1, version 3042169
      Extended community: Flags 0xe: SoO:1.1.1.2:100 EVPN MAC Mobility:0x00:2 EVPN ARP/ND:0x00
0x060e:0000.0000.0064 RT:100:100 RT:60100:100
      EVPN ESI: 0011.1111.1111.1111.1111
```

```
        PSID-Type:L3, SubTLV Count:1
         SID Locator: fccc:cc03:1031::/48, RIB metric: 0
         SubTLV:
          T:1(Sid information), Sid:fccc:cc03:1031::(Transposed), Behavior:63, SS-TLV Count:1
           SubSubTLV:
            T:1(Sid structure):
        PSID-Type:L2, SubTLV Count:1
         SID Locator: fccc:cc03:1031::/48, RIB metric: 0
         SubTLV:
          T:1(Sid information), Sid:fccc:cc03:1031::(Transposed), Behavior:67, SS-TLV Count:1
           SubSubTLV:
            T:1(Sid structure):
  Path #2: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    fccc:cc00:1102::1 (metric 20) from fccc:cc00:1301::1 (1.1.1.2)
      Received Label 0xe00000, Second Label 0xe0180
      Origin IGP, localpref 100, valid, internal, import-candidate, imported, rib-install
      Received Path ID 1, Local Path ID 0, version 0
      Extended community: SoO:1.1.1.2:100 Color:100 EVPN MAC Mobility:0x00:2 EVPN ARP/ND:0x00
0x060e:0000.0000.0064 RT:100:100 RT:60100:100
      Originator: 1.1.1.2, Cluster list: 1.1.1.5
      EVPN ESI: 0011.1111.1111.1111.1111
      PSID-Type:L3, SubTLV Count:1
       SID Locator: fccc:cc03:1032::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1032::(Transposed), Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      PSID-Type:L2, SubTLV Count:1
       SID Locator: fccc:cc03:1032::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1032::(Transposed), Behavior:67, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.2:100
```

Step 22 For SRv6 core, run the **show bgp**, **show route**, , and **show cef** commands to verify remote forwarding.

Example:

```
Router# show bgp l2vpn evpn rd  1.1.1.3:100 [2][0][48][0011.0100.0001][32][171.1.1.2]/136
BGP routing table entry for [2][0][48][0011.0100.0001][32][171.1.1.2]/136, Route Distinguisher:
1.1.1.3:100
Versions:
  Process           bRIB/RIB   SendTblVer
  Speaker            4790491      4790491
Last Modified: Nov 10 14:32:36.242 for 00:01:44
Paths: (2 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    fccc:cc00:1101::1 (metric 20) from fccc:cc00:1301::1 (1.1.1.1)
      Received Label 0xe00000, Second Label 0xe0220
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported,
rib-install
      Received Path ID 1, Local Path ID 1, version 4790309
      Extended community: SoO:1.1.1.2:100 Color:100 EVPN MAC Mobility:0x00:2 EVPN ARP/ND:0x00
0x060e:0000.0000.0064 RT:100:100 RT:60100:100
      Originator: 1.1.1.1, Cluster list: 1.1.1.5
      EVPN ESI: 0011.1111.1111.1111.1111
      PSID-Type:L3, SubTLV Count:1
       SID Locator: fccc:cc03:1031::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1031::(Transposed), Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      PSID-Type:L2, SubTLV Count:1
       SID Locator: fccc:cc03:1031::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1031::(Transposed), Behavior:67, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.1:100
  Path #2: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    fccc:cc00:1102::1 (metric 20) from fccc:cc00:1301::1 (1.1.1.2)
      Received Label 0xe00000, Second Label 0xe0180
      Origin IGP, localpref 100, valid, internal, import-candidate, imported, rib-install
      Received Path ID 1, Local Path ID 0, version 0
      Extended community: SoO:1.1.1.2:100 Color:100 EVPN MAC Mobility:0x00:2 EVPN ARP/ND:0x00
0x060e:0000.0000.0064 RT:100:100 RT:60100:100
      Originator: 1.1.1.2, Cluster list: 1.1.1.5
      EVPN ESI: 0011.1111.1111.1111.1111
      PSID-Type:L3, SubTLV Count:1
       SID Locator: fccc:cc03:1032::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1032::(Transposed), Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      PSID-Type:L2, SubTLV Count:1
       SID Locator: fccc:cc03:1032::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1032::(Transposed), Behavior:67, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.2:100
```

Example:

```
Router# show bgp vpnv4 unicast vrf CE100 171.1.1.2/32
BGP routing table entry for 171.1.1.2/32, Route Distinguisher: 102:100
Versions:
  Process           bRIB/RIB   SendTblVer
  Speaker            1345462      1345462
Last Modified: Nov 10 14:32:36.242 for 00:01:51
Paths: (2 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    fccc:cc00:1101::1 (metric 20) from fccc:cc00:1301::1 (1.1.1.1)
      Received Label 0xe0220
      Origin IGP, localpref 100, valid, internal, best, group-best, multipath, import-candidate,
imported
      Received Path ID 1, Local Path ID 1, version 1345378
      Extended community: SoO:1.1.1.2:100 Color:100 EVPN MAC Mobility:0x00:2 EVPN ARP/ND:0x00
0x060e:0000.0000.0064 RT:100:100 RT:60100:100
      mac: 00:11:01:00:00:01
      ethernet tag: 0
      Originator: 1.1.1.1, Cluster list: 1.1.1.5
      PSID-Type:L3, SubTLV Count:1
       SID Locator: fccc:cc03:1031::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1031::(Transposed), Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.1:100
  Path #2: Received by speaker 0
  Not advertised to any peer
  Local, (received & used)
    fccc:cc00:1102::1 (metric 20) from fccc:cc00:1301::1 (1.1.1.2)
      Received Label 0xe0180
      Origin IGP, localpref 100, valid, internal, multipath, import-candidate, imported
      Received Path ID 1, Local Path ID 0, version 0
      Extended community: SoO:1.1.1.2:100 Color:100 EVPN MAC Mobility:0x00:2 EVPN ARP/ND:0x00
0x060e:0000.0000.0064 RT:100:100 RT:60100:100
      mac: 00:11:01:00:00:01
      ethernet tag: 0
      Originator: 1.1.1.2, Cluster list: 1.1.1.5
      PSID-Type:L3, SubTLV Count:1
       SID Locator: fccc:cc03:1032::/48, RIB metric: 210
       SubTLV:
        T:1(Sid information), Sid:fccc:cc03:1032::(Transposed), Behavior:63, SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.2:100
```

Example:

```
Router# show route vrf CE100 171.1.1.2/32
Routing entry for 171.1.1.2/32
  Known via "bgp 100", distance 200, metric 0, type internal
  Installed Nov 10 14:32:35.791 for 00:01:58
  Routing Descriptor Blocks
    fccc:cc00:1101::1, from fccc:cc00:1301::1, BGP multi path
      Nexthop in Vrf: "default", Table: "default", IPv6 Unicast, Table Id: 0xe0800000
      Route metric is 0, recursion-via-/48
    fccc:cc00:1102::1, from fccc:cc00:1301::1, BGP multi path
      Nexthop in Vrf: "default", Table: "default", IPv6 Unicast, Table Id: 0xe0800000
      Route metric is 0, recursion-via-/48
  No advertising protos.
```

Example:

```
Router# show cef vrf CE100 171.1.1.2/32
171.1.1.2/32, version 36883, SRv6 Headend, internal 0x5000001 0x30 (ptr 0x90a4f8d4) [1], 0x0 (0x0),
 0x0 (0x92a32670)
 Updated Nov 10 14:32:35.800
 Prefix Len 32, traffic index 0, precedence n/a, priority 3
  gateway array (0x8f5d2198) reference count 1000, flags 0x2010, source rib (7), 0 backups
              [1 type 3 flags 0x48441 (0x79f055f8) ext 0x0 (0x0)]
  LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
  gateway array update type-time 1 Nov 10 13:53:39.004
 LDI Update time Nov 10 13:53:39.004

  Level 1 - Load distribution: 0 1
  [0] via fccc:cc03:1031::/128, recursive
  [1] via fccc:cc03:1032::/128, recursive

   via fccc:cc03:1031::/128, 14 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 0 NHID 0x0 [0x909d1ef4 0x0]
    recursion-via-/48
    next hop VRF - 'default', table - 0xe0800000
    next hop fccc:cc03:1031::/128 via fccc:cc03:1031::/48
    SRv6 H.Encaps.Red SID-list {fccc:cc03:1031:e022::}

    Load distribution: 0 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   Bundle-Ether1             fe80::28a:96ff:fe4a:20e1

   via fccc:cc03:1032::/128, 21 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 1 NHID 0x0 [0x909dd474 0x0]
    recursion-via-/48
    next hop VRF - 'default', table - 0xe0800000
    next hop fccc:cc03:1032::/128 via fccc:cc03:1032::/48
    SRv6 H.Encaps.Red SID-list {fccc:cc03:1032:e018::}

    Load distribution: 0 (refcount 1)

    Hash  OK  Interface                 Address
    1     Y   Bundle-Ether1             fe80::28a:96ff:fe4a:20e1
```

Step 23 Verify the subnet stretching.

a) Run the **show run vrf cust130** command to verify the VRF configuration.

Example:

```
Router# show run vrf cust130
vrf cust130
address-family ipv4 unicast
 import route-target
  130:130
 !
 export route-target
  130:130
 !
!
!
```

b) Run the **show run router bgp** command to verify the BGP configuration.

Example:

For MPLS core:

```
Router# show run router bgp | begin vrf cust130
vrf cust130
 rd auto
 address-family ipv4 unicast
  label mode per-vrf
  maximum-paths ibgp 10
  redistribute connected
 !
!
```

Example:

For SRv6 core:

```
Router# show run router bgp 200 vrf cust130
router bgp 200
 vrf cust130
  rd auto
  address-family ipv4 unicast
   maximum-paths ibgp 10
   segment-routing srv6
    locator loc_FA135
    alloc mode per-vrf
   redistribute connected
   !
  !
 !
```

c) Run the **show run l2vpn bridge group** command to verify the L2VPN configuration.

Example:

```
Router# show run l2vpn bridge group bg130
l2vpn
bridge group bg130
 bridge-domain bd130
  interface Bundle-Ether1.1300
  !
  interface Bundle-Ether5.1300
  !
  routed interface BVI130
  evi 130
  !
 !
!
!
```

EVPN IPv6 Hosts with Mobility

EVPN IPv6 Hosts with Mobility feature enables you to provide EVPN IPv6 service over IPv4-MPLS core network. This feature supports all-active multihoming and virtual machine (VM) or host move.

Table 6: Feature History Table

Feature Name	Release Information	Feature Description
EVPN IPv6 Hosts with Mobility over SRv6 core	Release 25.4.1	You can enable the EVPN IPv6 Hosts with Mobility feature to provide IPv6 EVPN services with host mobility and all-active multihoming over an SRv6 core. This enables service providers to deliver IPv6 VPN services using a programmable, flexible SRv6 backbone, natively supporting IPv6 reachability and segment routing-based label distribution for seamless IPv6 service delivery.
EVPN IPv6 Hosts with Mobility over MPLS core	Release 6.3.2	You can enable the EVPN IPv6 Hosts with Mobility feature to provide IPv6 EVPN services with host mobility and all-active multihoming over an IPv4 MPLS core. This allows service providers to deliver IPv6 VPN services using a stable IPv4-MPLS backbone, leveraging technologies like 6PE and 6VPE for IPv6 reachability and label distribution without requiring a native IPv6 core.

Service Providers (SPs) use a stable and established core with IPv4-MPLS backbone for providing IPv4 VPN services. The IPv6 VPN Provider Edge Transport over MPLS (IPv6 on Provider Edge Routers [6PE] and IPv6 on VPN Provider Edge Routers [6VPE]) facilitates SPs to offer IPv6 VPN services over IPv4 backbone without an IPv6 core. The provide edge (PE) routers run MP-iBGP to advertise IPv6 reachability and IPv6 label distribution. For 6PE, the labels are allocated per IPv6 prefix learnt from connected customer edge (CE) routers and for 6VPE, the PE router can be configured to allocate labels on a per-prefix or per-CE and per-VRF level.

Mobility Support

In global VRF, mobility is not supported. However, you can move a host from one ES to another ES within the same bridge domain. The host gets a new MAC address and IP address. The host can have multiple IP addresses for the same MAC address.

In non-default VRF, mobility is supported with the following conditions:

- Basic MAC move: The IP address and MAC address remains the same. You can move a host from one ES to another ES with the same IP address and MAC address.
- Same MAC address but with a different IP address: The host gets a new IP address
- Same IP address but with a different MAC address: The host gets a new MAC address but retains the same IP address
- Multiple IP addresses with the same MAC address: Many VMs are involved in the same the MAC move

Restrictions

- In customer VRFs, when host routing is not configured, MAC-IP advertisement is different between zero ESI and none-zero ESI. When host routing is not configured, MAC-IP with non-zero ESI is advertised without L3 RT (VRF RT). MAC-IP with zero ESI is not advertised. The following table lists the behavior of MAC-IP advertisement with respect to ESI and host routing.

ESI Type	With host routing	Without host routing
MAC-IP with non-zero ESI	Advertised with L3 VRF RT	Advertised without L3 VRF RT
MAC-IP with zero ESI	Advertised with L3 VRF RT	Not advertised

- In global VRF, Layer 2 stretch is not supported.
- MAC move in global VRF is only supported if the host is within the same bridge domain. You can move a host from one ES to another ES within the same bridge domain.
- Duplication of IP address detection is not supported.
- Maximum number of leafs allowed per ESI is two.

Configure EVPN IPv6 hosts with mobility

If you are deploying over an SRv6 core, make sure you configure SRv6 underlay on the participating routers.

Perform the following tasks to configure EVPN IPv6 Hosts with Mobility feature:

- Configure VRF
- Configure ISIS
- Configure BGP
- Configure AC interface
- Configure BVI interface
- Configure EVPN

- Configure L2VPN

Note

- You cannot configure the EVPN remote peer using the VPNv4 unicast if you have configured the **advertise vpnv4 unicast re-originated** command under the L2VPN EVPN address-family. You can either configure the VPNv4 unicast or the advertise vpnv4 unicast re-originated under L2VPN EVPN address-family.
- You cannot configure the EVPN remote peer using the VPNv6 unicast if you have configured the **advertise vpnv6 unicast re-originated** command under the L2VPN EVPN address-family. You can either configure the VPNv6 unicast or the advertise vpnv6 unicast re-originated under L2VPN EVPN address-family.

Before you begin

BGP label mode requirement for IPv6 default VRF IRB:

- From Release 6.5.1, IPv6 IRB in the default VRF (global routing table) requires per-VRF label allocation for the IPv6 BGP address family.
- You must configure **label mode per-vrf** under:
    - **router bgp address-family ipv6 unicast** for global or default VRF.
    - **router bgp vrf <vrf-name> address-family ipv6 unicast** if other VRFs are used alongside default VRF.

  Without **label mode per-vrf**, EVPN Type 2 re-originated IPv6 host routes (/128) and Type 5 IPv6 prefix routes are not installed with the correct VPN label, leading to routing failures and potential traffic loss for IPv6 IRB in the default VRF.

Perform these steps for EVPN IPv6 hosts with mobility. You can perform this task over an MPLS core, and from Release 25.4.1, you can perform this task over an SRv6 core.

Procedure

Step 1 Configure VRF with IPv4 and IPv6 address family to define the Virtual Routing and Forwarding instance and its import/export route targets for both IPv4 and IPv6 unicast.

Example:

```
Router# configure
Router(config)# vrf cust102
Router(config-vrf)# address-family ipv4 unicast
Router(config-vrf-af)# import route-target 160102:16102
Router(config-vrf-af)# export route-target 160102:16102
Router(config-vrf-af)# exit
Router(config-vrf)# address-family ipv6 unicast
Router(config-vrf-af)# import route-target 6160102:16102
Router(config-vrf-af)# export route-target 6160102:16102
Router(config-vrf-af)# commit
```

Step 2 For SRv6 core, configure SRv6 within a specific VRF address-family to enable per-VRF SID allocation for SRv6.

Example:

```
Router(config-vrf-af)# segment-routing srv6
Router(config-vrf-af-srv6)# alloc mode per-vrf
Router(config-vrf-af-srv6)# exit
Router(config-vrf-af)# commit
```

Step 3 Configure the Intermediate System to Intermediate System (ISIS) routing protocol for IPv6 to enable ISIS routing.

Example:

```
Router# configure
Route(config)# router isis v6
Route(config-isis)# 49.0001.0000.0160.0005.00
Route(config-isis)# nsr
Route(config-isis)# log adjacency changes
Route(config-isis)# lsp-gen-interval maximum-wait 5000 initial-wait 1 secondary-wait 20
Route(config-isis)# lsp-mtu 1468
Route(config-isis)# lsp-refresh-interval 65000
Route(config-isis)# max-lsp-lifetime 65535
```

Step 4 Configure MPLS and classic SR for ISIS.

Example:

For MPLS core:

```
Route(config-isis)# address-family ipv4 unicast
Route(config-isis-af)# metric-style wide
Route(config-isis-af)# microloop avoidance protected
Route(config-isis-af)# spf-interval maximum-wait 5000 initial-wait 1 secondary-wait 20
Route(config-isis-af)# segment-routing mpls sr-prefer
Route(config-isis-af)# segment-routing prefix-sid-map advertise-local
Route(config-isis-af)# exit
```

Example:

For SRv6 core:

```
Route(config-isis)# address-family ipv6 unicast
Route(config-isis-af)# metric-style wide
Route(config-isis-af)# segment-routing srv6
Route(config-isis-af-srv6)# locator SRV6_LOC1
Route(config-isis-af-srv6)# exit
Route(config-isis-af)# exit
```

Step 5 Configure ISIS loopback interface to make the loopback interface passive and enable ISIS for the respective address family.

Example:

For MPLS core:

```
Route(config-isis)# interface Bundle-Ether10
Route(config-isis-if)# point-to-point
Route(config-isis-if)# address-family ipv4 unicast
Route(config-isis-af)# fast-reroute per-prefix
Route(config-isis-af)# fast-reroute per-prefix ti-lfa
Route(config-isis-af)# metric 10
Route(config-isis-af)# exit
Route(config-isis)# interface Bundle-Ether20
Route(config-isis-if)# point-to-point
Route(config-isis-if)# address-family ipv4 unicast
Route(config-isis-af)# fast-reroute per-prefix
Route(config-isis-af)# fast-reroute per-prefix ti-lfa
Route(config-isis-af)# metric 10
Route(config-isis-af)# exit
```

Example:

For SRv6 core:

```
Route(config-isis)# interface Bundle-Ether10
Route(config-isis-if)# point-to-point
Route(config-isis-if)# address-family ipv6 unicast
Route(config-isis-af)# fast-reroute per-prefix
Route(config-isis-af)# fast-reroute per-prefix ti-lfa
Route(config-isis-af)# metric 10
Route(config-isis-af)# exit
Route(config-isis)# interface Bundle-Ether20
Route(config-isis-if)# point-to-point
Route(config-isis-if)# address-family ipv6 unicast
Route(config-isis-af)# fast-reroute per-prefix
Route(config-isis-af)# fast-reroute per-prefix ti-lfa
Route(config-isis-af)# metric 10
Route(config-isis-af)# exit
```

Step 6 Configure ISIS loopback interface to make the loopback interface passive and enable ISIS for the respective address family.

Example:

For MPLS core, configure ISIS loopback interface with IPv4 address-family.

```
Route(config-isis)# interface loopback0
Route(config-isis-if)# passive
Route(config-isis-if)# address-family ipv4 unicast
Route(config-isis-af)# exit
```

Example:

For Srv6 core, configure ISIS loopback interface with IPv6 address-family.

```
Route(config-isis-if)# address-family ipv6 unicast
Route(config-isis-af)# exit
```

Step 7 Configure explicit prefix SID to assign a prefix SID for the specified loopback interface and address family.

Example:

For MPLS core, configure explicit prefix SID for IPv4 address family.

```
Route(config-isis)# interface loopback10
Route(config-isis-if)# passive
Route(config-isis-if)# address-family ipv4 unicast
Route(config-isis-af)# prefix-sid index 1605
Route(config-isis-af)# exit
```

Example:

For SRv6 core, configure explicit prefix SID for IPv6 address family.

```
Route(config-isis-if)# address-family ipv6 unicast
Route(config-isis-af)# segment-routing srv6
Route(config-isis-af-srv6)# locator SRV6_LOC1
Route(config-isis-af-srv6)# exit
```

Step 8 Configure the global SR to define the global block for Segment Routing.

Example:

```
Router# configure
Router(config)# segment-routing
Router(config-sr)# global-block 16000 23999
```

Step 9 For SRv6, configure SR global block to define the SRv6 locator and its prefix.

Example:

```
Router(config-sr)# srv6
Router(config-sr-srv6)# locator SRV6_LOC1
Router(config-sr-srv6-loc)# prefix 2001:db8:1000::/48
Router(config-sr-srv6-loc)# exit
Router(config-sr-srv6)# exit
Router(config-sr)# commit
```

Step 10 Perform these steps to configure BGP:

a) Enable IPv4 Global VRF support and ECMP (Equal-Cost Multi-Path) with unequal-cost paths.

Example:

For MPLS core:

```
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# bfd minimum-interval 50
Router(config-bgp)# bfd multiplier 3
Router(config-bgp)# bgp router-id 160.0.0.5
Router(config-bgp)# address-family ipv4 unicast ---> To support V4 Global VRF
Router(config-bgp-af)# maximum-paths ibgp 10 unequal-cost ---> ECMP
Router(config-bgp-af)# redistribute connected --> V4 Global VRF
Router(config-bgp-af)# exit
```

Example:

For SRv6 core:

```
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# bgp router-id 160.0.0.5
Router(config-bgp)# address-family ipv4 unicast ---> To support V4 Global VRF
Router(config-bgp-af)# maximum-paths ibgp 10 ---> ECMP
Router(config-bgp-af)# segment-routing srv6
Router(config-bgp-af-srv6)# locator uSID_LOC1
Router(config-bgp-af-srv6)# alloc mode per-vrf
Router(config-bgp-af-srv6)# exit
Router(config-bgp-af)# redistribute connected --> V4 Global VRF
Router(config-bgp-af)# exit
```

b) Configure BGP for a VRF instance to enable BGP for all VRFs and set the label mode.

Example:

For MPLS core:

```
Router(config-bgp)# address-family ipv4 unicast ---> VRF
Router(config-bgp-af)# vrf all
Router(config-bgp-af)# label mode per-vrf
Router(config-bgp-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp)# address-family vpnv4 unicast ---> VRF
Router(config-bgp-af)# vrf all
Router(config-bgp-af)# segment-routing srv6
Router(config-bgp-af-srv6)# locator uSID_LOC1
Router(config-bgp-af-srv6)# alloc mode per-vrf
Router(config-bgp-af-srv6)# exit
Router(config-bgp-af)# exit
```

c) Enable IPv6 routing to configure BGP for IPv6 unicast, set maximum paths, and redistribute static or connected routes.

Example:

For MPLS core:

```
Router(config-bgp)# address-family ipv6 unicast ---> For 6PE
Router(config-bgp-af)# label mode per-vrf
Router(config-bgp-af)# maximum-paths ibgp 8
Router(config-bgp-af)# redistribute static
Router(config-bgp-af)# allocate-label all
Router(config-bgp-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp)# address-family ipv6 unicast
Router(config-bgp-af)# maximum-paths ibgp 10
Router(config-bgp-af)# segment-routing srv6
Router(config-bgp-af-srv6)# locator uSID_LOC1
Router(config-bgp-af-srv6)# alloc mode per-vrf
Router(config-bgp-af-srv6)# exit
Router(config-bgp-af)# redistribute connected
Router(config-bgp-af)# exit
```

d) Configure BGP for VPNv6 services to enable VPNv6 unicast for all VRFs.

Example:

For MPLS core:

```
Router(config-bgp)# address-family vpnv6 unicast ---> 6 VPE
Router(config-bgp-af)# vrf all
Router(config-bgp-af)# label mode per-vrf
Router(config-bgp-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp)# address-family vpnv6 unicast ---> 6 VPE
Router(config-bgp-af)# vrf all
Router(config-bgp-af-vrf)# segment-routing srv6
Router(config-bgp-af-vrf-srv6)# locator uSID_LOC1
Router(config-bgp-af-vrf-srv6)# alloc mode per-vrf
Router(config-bgp-af-vrf-srv6)# exit
Router(config-bgp-af-vrf)# exit
Router(config-bgp-af)# exit
```

e) Enable BGP for EVPN services to activate the L2VPN EVPN address family and enable implicit import for Global VRF.

Example:

```
Router(config-bgp)# address-family l2vpn evpn ----> EVPN
Router(config-bgp-af)# bgp implicit-import ----> Global VRF
Router(config-bgp-af)# exit
```

f) Configure BGP neighbor-group for IPv4 unicast peering to define a neighbor group with remote AS, BFD settings, update source, and route policies for IPv4 unicast.

Example:

For MPLS core:

```
Router(config-bgp)# neighbor-group evpn-rr
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# update-source loopback0
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp)# neighbor-group evpn-rr
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# update-source loopback0
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# soft-reconfiguration inbound always
Router(config-bgp-nbr-af)# exit
```

g) Configure BGP neighbor to exchange IPv6 labeled unicast routes.

Example:

For MPLS core

```
Router(config-bgp-nbr)# address-family ipv6 labeled-unicast ----> For 6PE
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp-nbr)# address-family ipv6 unicast ----> For 6PE
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# soft-reconfiguration inbound always
Router(config-bgp-nbr-af)# exit
```

h) onfigure BGP neighbor for EVPN and interoperability with VPNv4 or VPNv6 unicast routes to enable the L2VPN EVPN address family for the neighbor, apply route policies, and optionally advertise re-originated VPNv4/VPNv6 unicast routes.

Example:

For MPLS core:

```
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# advertise vpnv4 unicast re-originated -> For Route Type 5
Router(config-bgp-nbr-af)# advertise vpnv6 unicast re-originated -> For Route Type 5
Router(config-bgp-nbr-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# soft-reconfiguration inbound always
Router(config-bgp-nbr-af)# exit
Router(config-bgp-nbr)# exit
```

i) Assign BGP neighbors to evpn-rr neighbor-group to apply the configurations

Example:

```
Router(config-bgp)# neighbor fc00:6400:106::1
Router(config-bgp-nbr)# use neighbor-group evpn-rr
Router(config-bgp-nbr)# exit
Router(config-bgp)# neighbor fc00:6400:107::1
Router(config-bgp-nbr)# use neighbor-group evpn-rr
Router(config-bgp-nbr)# exit
```

j) Apply BGP on VRF instances to configure BGP parameters specific to a VRF.

Example:

For MPLS core:

```
Router(config-bgp)# vrf all
Router(config-bgp-vrf)# rd 1605:102
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# label mode per-vrf
Router(config-bgp-vrf-af)# maximum-paths ibgp 10 unequal-cost
Router(config-bgp-vrf-af)# redistribute connected
Router(config-bgp-vrf-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp)# vrf CE1
Router(config-bgp-vrf)# rd auto
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# maximum-paths ibgp 10
Router(config-bgp-vrf-af)# segment-routing srv6
Router(config-bgp-vrf-af-srv6)# alloc mode per-vrf
Router(config-bgp-vrf-af-srv6)# exit
Router(config-bgp-vrf-af)# redistribute connected
Router(config-bgp-vrf-af)# redistribute static
Router(config-bgp-vrf-af)# exit
```

Configuring the **redistribute connected** triggers Route Type 5.

k) Configure BGP for IPv6 unicast routing within a VRF to enable IPv6 unicast for the VRF.

Example:

For MPLS core:

```
Router(config-bgp-vrf)# address-family ipv6 unicast
Router(config-bgp-vrf-af)# label mode per-vrf
Router(config-bgp-vrf-af)# maximum-paths ibgp 10 unequal-cost
Router(config-bgp-vrf-af)# redistribute connected
Router(config-bgp-vrf-af)# exit
```

Example:

For SRv6 core:

```
Router(config-bgp-vrf)# address-family ipv6 unicast
Router(config-bgp-vrf-af)# maximum-paths ibgp 10
Router(config-bgp-vrf-af)# segment-routing srv6
Router(config-bgp-vrf-af-srv6)# alloc mode per-vrf
Router(config-bgp-vrf-af-srv6)# exit
Router(config-bgp-vrf-af)# redistribute connected
Router(config-bgp-vrf-af)# redistribute static
Router(config-bgp-vrf-af)# exit
Router(config-bgp-vrf)# exit
Router(config-bgp)# exit
```

Step 11 Configure Layer 2 AC interfaces to enable Layer 2 transport on bundle sub-interfaces.

Example:

```
Router(config)# interface Bundle-Ether1.102 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 102
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit
Router(config-l2vpn-subif)# exit
```

Step 12 Configure BVI interface to assign IPv4 and IPv6 addresses.

Example:

```
Router(config)# interface BVI100
Router(config-if)# ipv4 address 56.78.100.1 255.255.255.0
Router(config-if)# ipv6 address 56:78:100::1/64
Router(config-if)# mac-address 22.22.22
Router(config-if)# exit
Router(config)# interface BVI102
Router(config-if)# host-routing
Router(config-if)# vrf cust102
Router(config-if-vrf)# ipv4 address 56.78.102.1 255.255.255.0
Router(config-if-vrf)# ipv6 nd dad attempts 0
Router(config-if-vrf)# ipv6 address 56:78:100::1/64
Router(config-if-vrf)# ipv6 address 56:78:102::1/64
Router(config-if-vrf)# mac-address 22.22.22
Router(config-if)# commit
```

Step 13 Configure EVPN to define the EVPN instance (EVI).

Example:

For MPLS core:

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 102
```

Example:

For SRv6 core:

```
Router# configure
Router(config)# evpn
Router(config-evpn)#  evi 102 segment-routing srv6
```

Step 14 Configure main bundle ethernet segment parameters in EVPN to set the Route Distinguisher (RD).

Example:

```
Router(config-evpn-evi)# bgp
Router(config-evpn-evi)# rd 1605:102
Router(config-evpn-evi-bgp)# route-target import 160102:102
Router(config-evpn-evi-bgp)# route-target export 160102:102
Router(config-evpn-evi-bgp)# exit
Router(config-evpn-evi)# advertise-mac
Router(config-evpn-evi)# exit
```

Step 15 Configure L2VPN to define bridge groups and bridge domains.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg102
Router(config-l2vpn-bg)# bridge-domain bd102
Router(config-l2vpn-bg-bd)# interface Bundle-Ether1.102
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# interface Bundle-Ether2.102
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# interface Bundle-Ether3.102
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# interface Bundle-Ether4.102
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# interface Bundle-Ether5.102
Router(config-l2vpn-bg-bd-ac)# routed interface BVI102
```

Step 16 Run the **show bgp ipv6** command to verify the host route

Example:

In the following example host route is advertised as EVPN route type 2.

```
Router# show bgp ipv6 unicast 56:78:100::2
BGP routing table entry for 56:78:100::2/128
Versions:
Process bRIB/RIB SendTblVer
Speaker 212 212
Local Label: 2
Last Modified: Oct 31 19:13:10.998 for 00:00:19
Paths: (1 available, best #1)
Not advertised to any peer
Path #1: Received by speaker 0
Not advertised to any peer
Local
160.5.5.5 (metric 20) from 160.0.0.1 (160.0.0.5)
Received Label 2
Origin IGP, localpref 100, valid, internal, best, group-best, imported
Received Path ID 0, Local Path ID 0, version 212
Extended community: Flags 0x20: SoO:160.5.5.5:100 RT:160100:100
mac: 00:06:01:00:01:02
Originator: 160.0.0.5, Cluster list: 100.0.0.4
Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1605:100
```

Step 17 Run the **show bgp ipv6** command to verify manually configured static routes in the global VRF.

Example:

For MPLS core:

```
Router# show bgp ipv6 unicast 56:78:100::2
BGP routing table entry for 30::1/128
Versions:
Process bRIB/RIB SendTblVer
Speaker 9 9
Local Label: 2
Last Modified: Oct 30 20:25:17.159 for 23:15:55
Paths: (2 available, best #2)
Advertised to update-groups (with more than one peer):
0.2
```

```
Path #1: Received by speaker 0
Not advertised to any peer
Local
160.0.0.6 (metric 20) from 160.0.0.1 (160.0.0.6)
Received Label 2
Origin incomplete, metric 0, localpref 100, valid, internal, labeled-unicast
Received Path ID 0, Local Path ID 0, version 0
mac: 10:11:04:64:f2:7f
Originator: 160.0.0.6, Cluster list: 100.0.0.4
Path #2: Received by speaker 0
Advertised to update-groups (with more than one peer): 0.2
Local
56:78:100::2 from :: (160.0.0.5)
Origin incomplete, metric 0, localpref 100, weight 32768, valid, redistributed, best,
group-best
Received Path ID 0, Local Path ID 0, version 9
mac: 10:11:04:64:f2:7f
```

Example:

For SRv6 core:

```
Router# show bgp ipv6 unicast 1000:64::23/128 detail
BGP routing table entry for 1000:64::23/128
Versions:
  Process           bRIB/RIB   SendTblVer
  Speaker                 28           28
    Flags: 0x0006b220+0x20010000+0x00000000 multipath;
Last Modified: Oct 21 19:31:56.614 for 00:01:27
Paths: (6 available, best #3)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Flags: 0x2000000000020005+0x00, import: 0x020
  Not advertised to any peer
  Local
    fc00:6400:102::1 (metric 21) from fc00:6400:106::1 (1.1.1.2), if-handle 0x00000000
      Origin IGP, localpref 100, valid, internal
      Received Path ID 1, Local Path ID 0, version 0
      Extended community: Flags 0x20: SoO:1.1.1.2:100 EVPN ARP/ND:0x00 0x060e:0000.0000.0064
RT:100:100
      Originator: 1.1.1.2, Cluster list: 1.1.1.6
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SID Locator RIB metric: 21 (loc_nh=0x7f5a5bc6b0b0)
       SubTLV:
        T:1(Sid information), Sid:fc00:6400:102:e037::, F:0x00, R2:0x00, Behavior:62, R3:0x00,
SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
  Path #2: Received by speaker 0
  Flags: 0x2000000000070005+0x00, import: 0x020
  Not advertised to any peer
  Local
    fc00:6400:103::1 (metric 21) from fc00:6400:106::1 (1.1.1.3), if-handle 0x00000000
      Origin IGP, localpref 100, valid, internal, multipath, add-path
      Received Path ID 2, Local Path ID 2, version 25
      Extended community: Flags 0x20: SoO:1.1.1.2:100 EVPN ARP/ND:0x00 0x060e:0000.0000.0064
RT:100:100
      Originator: 1.1.1.3, Cluster list: 1.1.1.6
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SID Locator RIB metric: 21 (loc_nh=0x7f5a5bc66a60)
       SubTLV:
        T:1(Sid information), Sid:fc00:6400:103:e03e::, F:0x00, R2:0x00, Behavior:62, R3:0x00,
SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
  Path #3: Received by speaker 0
  Flags: 0x2601000001070005+0x00, import: 0x0a0
  Not advertised to any peer
  Local
    fc00:6400:102::1 (metric 21) from fc00:6400:106::1 (1.1.1.2), if-handle 0x00000000
      Received Label 3
      Origin IGP, localpref 100, valid, internal, best, group-best, multipath, imported
      Received Path ID 0, Local Path ID 1, version 26
      Extended community: Flags 0x20: SoO:1.1.1.2:100 EVPN ARP/ND:0x00 0x060e:0000.0000.0064
RT:100:100
      mac: 00:23:00:00:00:64
      ethernet tag: 0
      Originator: 1.1.1.2, Cluster list: 1.1.1.6
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SID Locator RIB metric: 21 (loc_nh=0x7f5a5bc6b0b0)
       SubTLV:
        T:1(Sid information), Sid:fc00:6400:102:e037::, F:0x00, R2:0x00, Behavior:62, R3:0x00,
SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.2:100
  Path #4: Received by speaker 0
  Flags: 0x2601000000020005+0x00, import: 0x0a0
  Not advertised to any peer
  Local
    fc00:6400:103::1 (metric 21) from fc00:6400:106::1 (1.1.1.3), if-handle 0x00000000
      Received Label 3
      Origin IGP, localpref 100, valid, internal, imported
      Received Path ID 0, Local Path ID 0, version 0
      Extended community: Flags 0x20: SoO:1.1.1.2:100 EVPN ARP/ND:0x00 0x060e:0000.0000.0064
RT:100:100
      mac: 00:23:00:00:00:64
      ethernet tag: 0
      Originator: 1.1.1.3, Cluster list: 1.1.1.6
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SID Locator RIB metric: 21 (loc_nh=0x7f5a5bc66a60)
       SubTLV:
        T:1(Sid information), Sid:fc00:6400:103:e03e::, F:0x00, R2:0x00, Behavior:62, R3:0x00,
SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
      Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 1.1.1.3:100
  Path #5: Received by speaker 0
  Flags: 0x2000000000020005+0x00, import: 0x020
  Not advertised to any peer
  Local
    fc00:6400:102::1 (metric 21) from fc00:6400:107::1 (1.1.1.2), if-handle 0x00000000
      Origin IGP, localpref 100, valid, internal
      Received Path ID 1, Local Path ID 0, version 0
      Extended community: Flags 0x20: SoO:1.1.1.2:100 EVPN ARP/ND:0x00 0x060e:0000.0000.0064
RT:100:100
      Originator: 1.1.1.2, Cluster list: 1.1.1.7
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SID Locator RIB metric: 21 (loc_nh=0x7f5a5bc6b0b0)
       SubTLV:
        T:1(Sid information), Sid:fc00:6400:102:e037::, F:0x00, R2:0x00, Behavior:62, R3:0x00,
SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
  Path #6: Received by speaker 0
  Flags: 0x2000000000020005+0x00, import: 0x020
  Not advertised to any peer
  Local
    fc00:6400:103::1 (metric 21) from fc00:6400:107::1 (1.1.1.3), if-handle 0x00000000
      Origin IGP, localpref 100, valid, internal
      Received Path ID 2, Local Path ID 0, version 0
      Extended community: Flags 0x20: SoO:1.1.1.2:100 EVPN ARP/ND:0x00 0x060e:0000.0000.0064
RT:100:100
      Originator: 1.1.1.3, Cluster list: 1.1.1.7
      PSID-Type:L3, SubTLV Count:1, R:0x00,
       SID Locator RIB metric: 21 (loc_nh=0x7f5a5bc66a60)
       SubTLV:
        T:1(Sid information), Sid:fc00:6400:103:e03e::, F:0x00, R2:0x00, Behavior:62, R3:0x00,
SS-TLV Count:1
         SubSubTLV:
          T:1(Sid structure):
           Length [Loc-blk,Loc-node,Func,Arg]:[32,16,16,0], Tpose-len:0, Tpose-offset:0
```

Step 18 Run the **show evpn ethernet-segment** command to verify that ethernet segments are peering for dual homing.

Example:

```
Router# show evpn ethernet-segment int bundle-Ether 1
Ethernet Segment Id Interface Nexthops
------------------------ ---------------------------------- --------------------
0056.5656.5656.5656.5601 BE1 160.5.5.5
160.6.6.6
------------------------------------------------------------
```

Step 19 Run the **show evpn ethernet-segment** command to verify DF election.

Example:

```
Router# show evpn ethernet-segment int bundle-Ether 1 carving detail
Legend:
A - Load-balancing mode and Access Protection incompatible,
B - No Forwarders EVPN-enabled,
C - Backbone Source MAC missing (PBB-EVPN),
RT - ES-Import Route Target missing,
E - ESI missing,
H - Interface handle missing,
I - Name (Interface or Virtual Access) missing,
M - Interface in Down state,
O - BGP End of Download missing,
P - Interface already Access Protected,
Pf - Interface forced single-homed,
R - BGP RID not received,
S - Interface in redundancy standby state,
X - ESI-extracted MAC Conflict
SHG - No local split-horizon-group label allocated
Ethernet Segment Id Interface Nexthops
------------------------ ---------------------------------- --------------------
0056.5656.5656.5656.5601 BE1 160.5.5.5
160.6.6.6
ES to BGP Gates : Ready
ES to L2FIB Gates : Ready
Main port :
Interface name : Bundle-Ether1
Interface MAC : 008a.9644.acdd
IfHandle : 0x080004dc
State : Up
Redundancy : Not Defined
ESI type : 0
Value : 56.5656.5656.5656.5601
```

```
ES Import RT : 5656.5656.5656 (from ESI)
Source MAC : 0000.0000.0000 (N/A)
Topology :
Operational : MH
Configured : All-active (AApF) (default)
Primary Services : Auto-selection
Secondary Services: Auto-selection
Service Carving Results:
Forwarders : 161
Permanent : 10
EVI:ETag P : 700:1, 701:1, 702:1, 703:1, 704:1, 705:1
EVI:ETag P : 706:1, 707:1, 708:1, 709:1
Elected : 76
EVI E : 100, 102, 104, 106, 108, 110
EVI E : 112, 114, 116, 118, 120, 122,
EVI E : 124, 126, 128, 130, 132, 134,
EVI E : 136, 138, 140, 142, 144, 146,
EVI E : 148, 150, 152, 154, 156, 158,
EVI E : 160, 162, 164, 166, 168, 170,
EVI E : 172, 174, 176, 178, 180, 182,
EVI E : 184, 186, 188, 190, 192, 194,
EVI E : 196, 198, 200, 202, 204, 206,
EVI E : 208, 210, 212, 214, 216, 218,
EVI E : 220, 222, 224, 226, 228, 230,
EVI E : 232, 234, 236, 238, 240, 242,
EVI E : 244, 246, 248, 250
Not Elected : 75
EVI NE : 101, 103, 105, 107, 109, 111
EVI NE : 113, 115, 117, 119, 121, 123,
EVI NE : 125, 127, 129, 131, 133, 135,
EVI NE : 137, 139, 141, 143, 145, 147,
EVI NE : 149, 151, 153, 155, 157, 159,
EVI NE : 161, 163, 165, 167, 169, 171,
EVI NE : 173, 175, 177, 179, 181, 183,
EVI NE : 185, 187, 189, 191, 193, 195,
EVI NE : 197, 199, 201, 203, 205, 207,
EVI NE : 209, 211, 213, 215, 217, 219,
EVI NE : 221, 223, 225, 227, 229, 231,
EVI NE : 233, 235, 237, 239, 241, 243,
EVI NE : 245, 247, 249
MAC Flushing mode : STP-TCN
Peering timer : 3 sec [not running]
Recovery timer : 30 sec [not running]
Carving timer : 0 sec [not running]
Local SHG label : 68663
Remote SHG labels : 1
68670 : nexthop 160.6.6.6
```

EVPN IRB: DHCPv4 and DHCPv6 Relay

EVPN IRB: DHCPv4 and DHCPv6 Relay feature provides DHCP support for the end users in EVPN multi-homing Active-Active (MH-AA) deployment scenario. This feature enables reduction of traffic flooding, increase in load sharing at VTEP, faster convergence during link and device failures, and simplification of data center automation.

DHCPv4 and DHCPv6 Relay agents relay request packets, coming over the access interface, to external DHCPv4 and DHCPv6 server to request allocation of addresses (/32) and IANA (::/128) for the end user.

DHCPv4 and DHCPv6 Relay profiles are configured on BVI interfaces which relay DHCPv4 or DHCPv6 requests from Layer 2 (L2) attachment circuit (AC) to external DHCP servers for host IPv4 addresses (/32) and IANA (::128) IPv6 addresses.

This feature is compliant with RFC-6607.

Multi-homing Active-Active EVPN Gateways

Multi-homing Active-Active EVPN Gateways are configured with anycast IP address and MAC addresses. ASR 9000 devices have centralized L2//Layer 3 (L3) gateway. Based on native EVPN and MAC learning, IRB uses distributed anycast IP and anycast MAC address. Static clients are configured with anycast gateway address as the default gateway. DHCP client sends DHCP requests for IP addresses with BVI as the gateway. L2 access can be either single homing or multi-homing, Not all access protocols is supported with IRB. There may or may not be L2 stretch between DC centers. Internet gateway is also included for clients to access external network. No EVPN is configured on the Internet gateway.

EVPN IRB Route Distribution

In EVPN IRB DHCPv4 and DHCPv6, DHCP application processes and DHCP packet forwarding are independent of EVPN IRB L2 and L3 routing. There is no subscriber routing information with the stateless DHCP relay. But DHCP clients work similar to static clients in the EVPN core for L2 and L3 bridging and routing. When the **relay information option**, **relay information option vpn**, **relay information option von-mode cisco** and **relay information option von-mode rfc** commands are configured on the DHCP relay agent, the DHCP relay agent inserts the sub options of DHCP Option 82, such as subnet selection and VPN ID options. These options are considered by DHCP server while allocating IP address.

DHCP clients use the L2 AC interface to access EVPN bridge domain and use BVI interface as default gateway. So the clients must get the IP addresses from the DHCP server as in the same subnet of BVI interface.

DHCP Request Forwarding Path

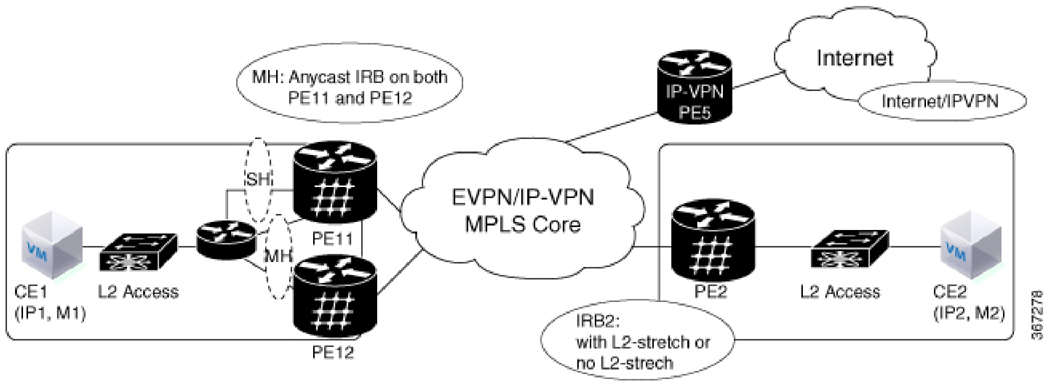
Clients broadcast requests to the access switch with DHAA to EVPN PE routers. The access switch does load balancing. The load balancing configurations in access switch will impact PE in DH-AA and DHCP to send the DHCP requests. The DHCP request reaches the Bridge Domain (BD) BVI interface which is configured with DHCP relay. Because AA PE routers are configured the same IP addresses, BVI IP addresses cannot be used as DHCP relay source IP address.

For DHCPv4, configuring GIADDR field for each DHCP relay profile is allowed. Loopback interface with unique IP addresses can be configured in VRF which is reachable to DHCP servers. Configuring DHCP relay source address is not supported.

In case of DHCPv6 servers, DHCPv6 relay picks up an available Loopback interface IPv6 address as DHCPv6 relay source IP address. After the DHCP clients get the IP address. DHCP clients are not normally routable to DHCP servers. DHCP clients send unicast DHCP renew messages to the DHCP server. If the DHCP servers are not routable, the DHCP unicast messages fail, then the DHCP client sends broadcast rebinding messages with the corrsponding DHCP relay.

In the below figure, DHCP clients are configured on PE11 and PE12.

Figure 5: EVPN IRB with ASR 9000 as Centralized DCI Gateway



Configure EVPN IRB: DHCPv4 and DHCPv6 Relay

Perform the following tasks to configure the EVPN IRB: DHCPv4 and DHCPv6 Relay feature:

```
/* PE11 configuration */

Router(config)# dhcp ipv4
Router(config-dhcpv4)# profile DHCPv4_RELAY relay
Router(config-dhcpv4-relay-profile)# helper-address vrf default 10.20.20.20 giaddr 192.0.2.1
Router(config-dhcpv4-relay-profile)# relay information option vpn
Router(config-dhcpv4-relay-profile)# relay information option vpn-mode rfc
Router(config-dhcpv4-relay-profile)# exit
Router(config-dhcpv4)# exit
Router(config)# interface BVI1 relay profile DHCPv4_RELAY


Router(config)# dhcp ipv6
Router(config-dhcpv6)# profile DHCPv6_RELAY relay
Router(config-dhcpv6-relay-profile)# helper-address vrf default 20::20
Router(config-dhcpv6-relay-profile)# exit
Router(config-dhcpv6)# exit
Router(config)# interface BVI1 relay profile  DHCPv6_RELAY
Router(config-if)# exit
Router(config)# interface Loopback 5
Router(config-if)# ipv4 address 192.0.2.1 255.255.255.255
Router(config-if)# exit
Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# ipv4 address 10.10.10.2 255.255.255.0
Router(config-if)# ipv6 address 2001:DB8:0:ABCD::1/64
Router(config-if)# ipv6 enable
Router(config-if)# mac-address 0.12.3456

/* PE12 configuration */
Router# dhcp ipv4
Router(config-dhcpv4)# profile DHCPv4_RELAY relay
Router(config-dhcpv4-relay-profile)# helper-address vrf default 10.20.20.20 giaddr 127.0.0.1
Router(config-dhcpv4-relay-profile)# relay information option vpn
Router(config-dhcpv4-relay-profile)# relay information option vpn-mode cisco
Router(config-dhcpv4-relay-profile)# exit
Router(config-dhcpv4)# exit
Router(config)# interface BVI1 relay profile DHCPv4_RELAY
Router(config)#  dhcp ipv6
Router(config-dhcpv6)# profile DHCPv6_RELAY relay
Router(config-dhcpv6-relay-profile)# helper-address vrf default 20::20
Router(config-dhcpv6-relay-profile)# exit
Router(config-dhcpv6)# exit
Router(config) interface BVI1 relay profile DHCPv6_RELAY
Router(config)# interface Loopback 6
Router(config)# exit
Router(config-if)# ipv4 address 127.0.0.1 255.255.255.255

Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# vrf evpn 1
Router(config-if)# exit
Router(config-if)# ipv4 address 10.10.10.2 255.255.255.0
Router(config-if)# proxy-arp
Router(config-if)# ipv6 address 3000:0:0:8003::2/64
Router(config-if)# ipv6 enable
Router(config-if)# mac-address 1122.3344.5566
```

Running Configuration

```
/* PE11 Configuration */

dhcp ipv4
profile DHCPv4_RELAY relay
  helper-address vrf default 10.20.20.20 giaddr 192.0.2.1
  relay information option vpn
relay information option vpn-mode cisco
!
interface BVI1 relay profile DHCPv4_RELAY
!
dhcp ipv6
profile DHCPv6_RELAY relay
  helper-address vrf default 20::20
!
interface BVI1 relay profile  DHCPv6_RELAY
!
interface Loopback5
ipv4 address 192.0.2.1 255.255.255.0
!
interface BVI1
host-routing
ipv4 address 10.10.10.2 255.255.255.0
ipv6 address  2001:DB8:0:ABCD::1/64
ipv6 enable
mac-address 0.12.3456
!

/* PE12 Configuration */

dhcp ipv4
profile DHCPv4_RELAY relay
  helper-address vrf default 10.20.20.20 giaddr 127.0.0.1
  relay information option vpn
  relay information option vpn-mode cisco
!
interface BVI1 relay profile DHCPv4_RELAY
!
dhcp ipv6
```

```
profile DHCPv6_RELAY relay
  helper-address vrf default 20::20
!
interface BVI1 relay profile DHCPv6_RELAY
!
interface Loopback6
ipv4 address 127.0.0.1 255.255.255.255
!
interface BVI1
host-routing
vrf evpn1
ipv4 address 10.10.10.2 255.255.255.0
proxy-arp
ipv6 address 3000:0:0:8003::2/64
ipv6 enable
mac-address 1122.3344.5566
!
```

Verification

Verify the DHCPv4 configuration.

```
Router# show running-configuration dhcp ipv4
Thu Feb 15 21:44:31.550 IST
dhcp ipv4
 profile DHCPv4_RELAY relay
  helper-address vrf default 10.20.20.20 giaddr 192.0.2.1
  relay information option vpn
  relay information option vpn-mode rfc
!
interface BVI1 relay profile DHCPv4_RELAY
!
```

Verify the DHCPv4 relay profile details.

```
Router# show dhcp ipv4 relay profile name DHCPv4_RELAY
Thu Feb 15 21:47:32.247 IST
Profile: DHCPv4_RELAY
Helper Addresses:
        10.20.20.20, vrf default, giaddr 192.0.2.1
Information Option: Disabled
Information Option Allow Untrusted: Disabled
Information Option VPN: Enabled
Information Option VPN Mode: RFC
Information Option Policy: Replace
Information Option Check: Disabled
GIADDR Policy: Keep
Broadcast-flag Policy: Ignore
Mac Mismatch Action: Forward
VRF References:
Interface References: BVI1
```

Verify the DHCPv4 relay packet statistics.

```
Router# show dhcp ipv4 relay statistics
```

```
Fri Feb 16 12:34:51.202 IST

      VRF                    |      RX   |    TX |   DR |
-------------------------------------------------------
default                      |      4    |    4  |   0  |
**nVSatellite                |      0    |    0  |   0  |
```

Verify DHCPv4 relay packet statistics in detail.

```
Router# show dhcp vrf default ipv4 relay statistics
Fri Feb 16 12:36:05.544 IST

DHCP IPv4 Relay Statistics for VRF default:

     TYPE          |    RECEIVE    |    TRANSMIT   |     DROP      |
-------------------------------------------------------------------
DISCOVER          |            1  |            1  |            0  |
OFFER             |            1  |            1  |            0  |
REQUEST           |            1  |            1  |            0  |
DECLINE           |            0  |            0  |            0  |
ACK               |            1  |            1  |            0  |
NAK               |            0  |            0  |            0  |
RELEASE           |            0  |            0  |            0  |
INFORM            |            0  |            0  |            0  |
LEASEQUERY        |            0  |            0  |            0  |
LEASEUNASSIGNED   |            0  |            0  |            0  |
LEASEUNKNOWN      |            0  |            0  |            0  |
LEASEACTIVE       |            0  |            0  |            0  |
BOOTP-REQUEST     |            0  |            0  |            0  |
BOOTP-REPLY       |            0  |            0  |            0  |
BOOTP-INVALID     |            0  |            0  |            0  |
```

Verify the DHCPv6 configuration.

```
Router# show running-configuration dhcp ipv6
Fri Feb 16 15:40:52.721 IST
dhcp ipv6
profile DHCPv6_RELAY relay
  helper-address vrf default 20::20
!
interface BVI1 relay profile DHCPv6_RELAY
!
```

Verify DHCPv6 relay profile.

```
Router# show dhcp ipv6 relay statistics
Fri Feb 16 15:41:00.456 IST

      VRF                    |      RX   |    TX |   DR |
-------------------------------------------------------------------------------------------
default                      |      4    |    4  |   0  |
**nVSatellite                |      0    |    0  |   0  |
```

Verify DHCPv6 relay packet statistics in detail.

```
Routerr# show dhcp ipv6 relay statistics vrf default
Fri Feb 16 15:41:09.991 IST

DHCP IPv6 Relay Statistics for VRF default:

     TYPE         |    RECEIVE     |    TRANSMIT    |      DROP      |
-------------------------------------------------------------------
SOLICIT          |            1  |            0  |            0  |
ADVERTISE        |            0  |            1  |            0  |
REQUEST          |            1  |            0  |            0  |
REPLY            |            0  |            1  |            0  |
CONFIRM          |            0  |            0  |            0  |
DECLINE          |            0  |            0  |            0  |
RENEW            |            0  |            0  |            0  |
REBIND           |            0  |            0  |            0  |
RELEASE          |            0  |            0  |            0  |
RECONFIG         |            0  |            0  |            0  |
INFORM           |            0  |            0  |            0  |
RELAY_FWD        |            0  |            0  |            0  |
RELAY_REP        |            0 |             0  |            0  |
LEASEQUERY       |            0  |            0  |            0  |
LEASEQUERY_REP   |            0  |            0  |            0  |
LEASEQUERY_DONE  |            0  |            0  |            0  |
LEASEQUERY_DATA  |            0  |            0  |            0  |
```

Duplicate IP Address Detection

The Duplicate IP Address Detection feature automatically detects any host with a duplicate IP address and blocks all MAC-IP routes that have a duplicate IP address.

This protects the network from hosts that are assigned duplicate IP addresses unintentionally or by malicious intent in an EVPN fabric. Hosts with duplicate IP address cause unnecessary churn in a network and causes traffic loss to either or both the hosts with the same IP address.

The system handles mobility of EVPN hosts by keeping track of MAC and IP addresses as they move from one host to another. If two hosts are assigned the same IP address, the IOS XR system keeps learning and re-learning MAC-IP routes from both the hosts. Each time it learns the MAC-IP route from one host, it is counted as one move since the newly learnt route supersedes the route previously learnt from the other host. This continues back and forth until the IP address is marked as duplicate based on the configured parameters.

It uses the following parameters to determine when an IP address should be marked as duplicate, and frozen or unfrozen as it moves between different hosts. The configurable parameters are:

- **move-interval**: The period within which a MAC or IP address has to move certain number of times between different hosts to be considered as duplicate and frozen temporarily. This number is specified in the **move-count** parameter.
- **move-count**: The number of times a MAC or IP address has to move within the interval specified for the **move-interval** parameter between different hosts to be considered a duplicate.
- **freeze-time**: The length of time a MAC or IP address is locked after it has been detected as a duplicate. After this period, the IP address is unlocked and it is allowed to learn again.
- **retry-count**: The number of times a MAC or IP address is unlocked after it has been detected as a duplicate before it is frozen permanently.

- **reset-freeze-count-interval**: The time window to count duplicate detection events before resetting the count.

The system maintains a count of the number of times an IP address has been moved from one host to another host, either to another local host or to a host behind a remote Top of Rack (TOR). If an IP address moves certain number of times specified in the **move-count** parameter within the interval specified in the **move-interval** parameter is considered a duplicate IP address. All MAC-IP routes with that IP address is frozen for the time specified in the **freeze-time** parameter. A syslog notifies the user that the particular IP address is frozen. While an IP address is frozen, any new MAC-IP routes or updates to existing MAC-IP routes with the frozen IP address are ignored.

After **freeze-time** has elapsed, the corresponding MAC-IP routes are unfrozen and the value of the **move-count** is reset to zero. For any unfrozen local MAC-IP routes, an ARP probe and flush are initiated while the remote MAC-IP routes are put in the probe mode. This restarts the duplicate detection process.

The system also maintains the information about the number of times a particular IP address has been frozen and unfrozen. If an IP address is marked as duplicate after it is unfrozen **retry-count** times, it is frozen permanently until user manually unfreezes it. Use the following commands to manually unfreeze frozen MAC, IPv4 and IPV6 addresses respectively:

- **clear l2route evpn mac**{*mac-address*} | **all** [**evi** *evi*] **frozen-flag**
- **clear l2route evpn ipv4**{*ipv4-address*} | **all** [**evi** *evi*] **frozen-flag**
- **clear l2route evpn ipv6**{*ipv6-address*} | **all** [**evi** *evi*] **frozen-flag**

Configure Duplicate IP Address Detection

Perfrom these tasks to configure Duplicate IP Address Detection feature.

Configuration Example

```
/* Ipv4 Address Duplicate Detection Configuration */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# host ipv4-address duplicate-detection
RP/0/RSP0/CPU0:router(config-evpn-host-ipv4-addr)# move-count 2
RP/0/RSP0/CPU0:router(config-evpn-host-ipv4-addr)# freeze-time 10
RP/0/RSP0/CPU0:router(config-evpn-host-ipv4-addr)# retry-count 2
RP/0/RSP0/CPU0:router(config-evpn-host-ipv4-addr)# commit

/* Ipv6 Address Duplicate Detection Configuration */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# host ipv6-address duplicate-detection
RP/0/RSP0/CPU0:router(config-evpn-host-ipv6-addr)# move-count 2
RP/0/RSP0/CPU0:router(config-evpn-host-ipv6-addr)# freeze-time 10
RP/0/RSP0/CPU0:router(config-evpn-host-ipv6-addr)# retry-count 2
RP/0/RSP0/CPU0:router(config-evpn-host-ipv6-addr)# commit
```

Running Configuration

This section shows the running configuration to detect duplicate IP address.

```
evpn
 host ipv4-address duplicate-detection
  move-count 2
  freeze-time 10
  retry-count 2
 !
evpn
 host ipv6-address duplicate-detection
  move-count 2
  freeze-time 10
  retry-count 2
 !
```

Verification

The show output given in the following section display the details of the duplicate IP address detection and recovery parameters.

```
RP/0/0/CPU0:leaf1# show l2route evpn mac-ip 2.1.1.10 detail
18:16:44 [1157/4613]
Flags: (Stt)=Static; (L)=Local; (Lp)=Local-Proxy; (R)=Remote;
     (N)=No Redistribution; (B)=Best Route;
     (P)=Probe; (S)=Peer Sync; (F)=Flush;
     (Dm)=Duplicate MAC; (Zm)=Frozen MAC;
     (Di)=Duplicate IP; (Zi)=Frozen IP;
     (Sfa)=Single Flow Active; (Spec)=Speculative;
     (A)=Access; (Gw)=Gateway; (I)=Immutable

Topo ID   Mac Address   IP Address Producer   Next Hop(s)             Seq No     Flags
Opaque Data Type Opaque Data Len
Opaque Data Value
Opaque NH Type Opaque NH Len
Opaque NH Value
-------- -------------- --------------- ----------- ---------------------------------------
 -------- --------
0         0010.0010.0010 2.1.1.10   LOCAL     Bundle-Ether10.212, N/A   7         BLDi
N/A           N/A
N/A
N/A           N/A
N/A

0         0011.0011.0011 2.1.1.10   L2VPN     28116/I/ME, N/A           8         BPDi
0             12
0x06000000 0x0110f0ff 0x00008000
N/A           N/A
N/A


RP/0/RSP0/CPU0:router# show l2route summary

Duplicate Detection Parameters
------ ---------- -------- ------- ---------- ---------- ----------------
Type   Disabled   Freeze   Move    Move       Retry      Freeze-Count
                  Time     Count   Interval   Count      Reset-Interval
------ ---------- -------- ------- ---------- ---------- ----------------
MAC    False      30       5       180        3          24
IPv4   False      30       5       180        3          24
IPv6   False      30       5       180        3          24
```

Related Topics

- Duplicate IP Address Detection, on page 60

Associated Commands

- evpn host ipv4-address duplicate-detection
- evpn host ipv6-address duplicate-detection
- show l2route evpn mac-ip all detail

EVPN E-Tree Using RT Constraints

The EVPN E-Tree using RT constraints feature enables you to configure BGP RT import and export policies for an attachment circuit. This feature allows you to define communication between the leaf and root nodes. The provider edge (PE) nodes can receive L2 traffic either from the attachment circuit (AC) of a bridge domain (BD) or from the remote PE node. For a given BD, L2 communication can only happen from root to leaf and leaf to root. This feature does not allow any L2 communication between the ACs of two or more leafs. This feature uses two BGP RTs for every EVI. Associate one RT with root ACs and the other with leaf ACs. For example, there are two distinct sets of RTs, one for root-rt and another for leaf-rt.

This feature provides you with the following benefits by performing filtering of unicast and multicast traffic at the ingress PE nodes:

- Achieve efficiency of the BGP MAC routes scale
- Reduce the consumption of hardware resources
- Utilize the link bandwidth efficiently

Rules for Import and Export Policies under the BGP of EVPN EVI Instances

- Root PE exports its ROOT-RT using BGP export policy. It also imports other ROOT-RT from the corresponding root PE for the same EVI. This is necessary where there is more than one root for a particular BD and EVPN EVI. For example, in a multihome active-active scenario or multihome port-active and single-active scenarios.
- Root PE imports LEAF-RT using BGP import policy for a EVPN EVI. This enables the root to be aware of all remote L2 MAC addresses through EVPN RT2 advertisement of leaf PE node for a given E-Tree EVI.
- Leaf PE exports its LEAF-RT using BGP export policy to let the root to be aware of the reachability of its directly connected L2 endpoints through EVPN RT2 advertisement.
- Leaf PE imports ROOT-RT using BGP import policy. It helps the leaf to know about the L2 endpoints which are reachable through the AC of BD under EVPN EVI instance of root PE. You must not import LEAF-RT using BGP Import policy to avoid L2 Communication between two leaf PEs.
- Use split-horizon filtering to block traffic among leaf ACs on a BD for a given E-Tree EVI.

The BGP import and export policies applies to all EVPN RTs along with the RT2 advertisement.

MAC Address Learning

- L2 MAC addresses are learnt on AC of a particular BD on leaf PE as type LOCAL. The same MAC address is advertised to root PE as EVPN RT2. On the remote root PE, the MAC table replicates the entry of MAC address with the learn type as L2VPN. Also, it associates the MPLS label of its BGP peer, which advertises RT2 to root PE node.
- L2 MAC addresses are learnt on AC of a particular BD on the root as type LOCAL. The same MAC address is advertised to peer root (except for MH A/A) or leaf PE as EVPN RT2. On the remote root PE or leaf PE, the MAC table replicates the entry of MAC address with the learn type as L2VPN. Also, it associates the MPLS label of its BGP peer, which advertises RT2 to PE node.
- L2 MAC addresses are learnt on AC of a particular BD on the root as type LOCAL. The same MAC address is advertised to peer root for MH A/A as EVPN RT2. The MAC table of the peer root node synchronizes the replicated entry of MAC address with the learn type as L2VPN for same the ESI and with the same AC as the next hop. This avoids flooding and duplication of known unicast traffic.

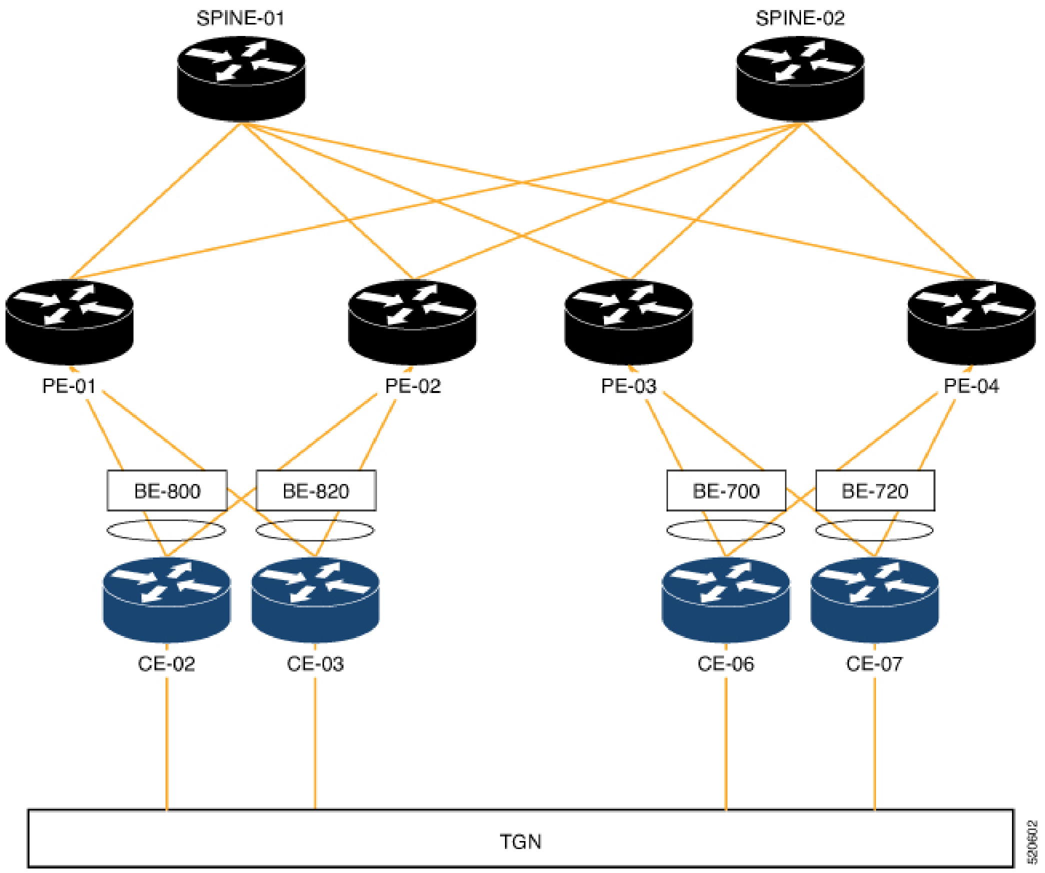
The following scenario describes the feature topology:

Limitation for EVPN E-Tree Using RT Constraints

- In Release 25.4.1, SRv6 core does not support the EVPN E-Tree per-PE.

CE with Multihoming Active-Active and CE with Multihoming Active-Active

Consider a topology where you connect CE-02 and CE-03 to PE-01 and PE-02. Both the CEs are in multihoming active-active mode. Connect CE-02 to PE-01 and PE-02 using AC BE-800.305. Connect CE-03 to PE-01 and PE-02 using AC BE-820.305. Connect CE-06 and CE-07 to PE-03 and PE-04. Connect CE-06 to PE-03 and PE-04 using AC BE-700.305. Connect CE-07 to PE-03 and PE-04 using AC BE-720.305. Associate the bridge domain BD-305 with other AC on the respective PEs along with EVI-305 instance. Configure the respective RT on root and leaf with its import and export RTs for EVI-305. Configure PE-01 and PE-02 as root. Configure PE-03 and PE-04 as leaf.



Configuration

Perform the following tasks on PE-01, PE-02, PE-03, and PE-04.

- Configure bridge domain
- Configure attachment circuit
- Configure EVPN EVI
- Configure bundle Ethernet
- Configure EVPN interface

Note Use the **etree rt-leaf** command only if the leaf sites are in the EVPN all-active multihoming mode and not required for EVPN single homing mode.

Configuration Example

```
/* Configure PE-01 (as root) */

/* Configure bridge domain */
Router # configure
```

```
Router(config)# l2vpn
Router(config-l2vpn)# bridge group EVPN_BD
Router(config-l2vpn-bg)# bridge-domain evpn_bvi_305
Router(config-l2vpn-bg-bd)# interface Bundle-Ether800.305
Router(config-l2vpn-bg-bd-ac)# exit
Router (config-l2vpn-bg-bd)# interface Bundle-Ether820.305
Router(config-l2vpn-bg-bd-ac)# exit
Router (config-l2vpn-bg-bd)# evi 305
Router (config-l2vpn-bg-bd-evi)# commit

/* Configure attachment circuit */
Router# configure
Router(config)# interface Bundle-Ether800.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

Router# configure
Router(config)# interface Bundle-Ether820.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

/* Configure EVPN EVI */
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 305
Router(config-evpn-instance)# bgp
Router(config-evpn-instance-bgp)# route-target import 1001:305
Router(config-evpn-instance-bgp)# route-target export 1001:305
Router(config-evpn-instance-bgp)# route-target import 1001:5305
Router(config-evpn-instance-bgp)# exit
Router(config-evpn-instance)# control-word-disable
Router(config-evpn-instance)# advertise-mac
Router(config-evpn-instance-mac)# commit

/* Configure bundle Ethernet */
Router# configure
Router(config)# interface Bundle-Ether800
Router(config-if)# lacp system mac 00aa.aabb.2020
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit

Router# configure
Router(config)# interface Bundle-Ether820
Router(config-if)# lacp system mac 00aa.aabb.2222
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit

/* Configure EVPN interface */
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether800
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.88.88.88.88.88.88.88.00
Router(config-evpn-ac-es)# bgp route-target 0001.0000.0001
Router(config-evpn-ac-es)# commit

Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether820
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.88.88.88.88.88.88.88.20
Router(config-evpn-ac-es)# bgp route-target 0001.0000.0020
Router(config-evpn-ac-es)# commit


/* Configure PE-02 (as root) */

/* Configure bridge domain */
Router # configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group EVPN_BD
Router(config-l2vpn-bg)# bridge-domain evpn_bvi_305
Router(config-l2vpn-bg-bd)# interface Bundle-Ether800.305
Router(config-l2vpn-bg-bd-ac)# exit
Router (config-l2vpn-bg-bd)# interface Bundle-Ether820.305
Router(config-l2vpn-bg-bd-ac)# exit
Router (config-l2vpn-bg-bd)# evi 305
Router (config-l2vpn-bg-bd-evi)# commit

/* Configure attachment circuit */
Router# configure
Router(config)# interface Bundle-Ether800.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

Router# configure
Router(config)# interface Bundle-Ether820.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

/* Configure EVPN EVI */
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 305
Router(config-evpn-instance)# bgp
Router(config-evpn-instance-bgp)# route-target import 1001:305
Router(config-evpn-instance-bgp)# route-target export 1001:305
Router(config-evpn-instance-bgp)# route-target import 1001:5305
Router(config-evpn-instance-bgp)# exit
Router(config-evpn-instance)# control-word-disable
Router(config-evpn-instance)# advertise-mac
Router(config-evpn-instance-mac)# commit

/* Configure bundle Ethernet */
Router# configure
Router(config)# interface Bundle-Ether800
Router(config-if)# lacp system mac 00aa.aabb.2020
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit

Router# configure
Router(config)# interface Bundle-Ether820
Router(config-if)# lacp system mac 00aa.aabb.2222
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit
```

```
/* Configure EVPN interface */
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether800
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.88.88.88.88.88.88.88.00
Router(config-evpn-ac-es)# bgp route-target 0001.0000.0001
Router(config-evpn-ac-es)# commit

Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether820
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.88.88.88.88.88.88.88.20
Router(config-evpn-ac-es)# bgp route-target 0001.0000.0020
Router(config-evpn-ac-es)# commit


/* Configure PE-03 (as leaf) */

/* Configure bridge domain */
Router # configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group EVPN_BD
Router(config-l2vpn-bg)# bridge-domain evpn_bvi_305
Router(config-l2vpn-bg-bd)# interface Bundle-Ether700.305
Router(config-l2vpn-bg-bd-ac)# split-horizon group
Router (config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# interface Bundle-Ether720.305
Router(config-l2vpn-bg-bd-ac)# split-horizon group
Router (config-l2vpn-bg-bd-ac)# exit
Router (config-l2vpn-bg-bd)# evi 305
Router (config-l2vpn-bg-bd-evi)# commit

/* Configure attachment circuit */
Router# configure
Router(config)# interface Bundle-Ether700.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

Router# configure
Router(config)# interface Bundle-Ether720.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

/* Configure EVPN EVI */
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 305
Router(config-evpn-instance)# bgp
Router(config-evpn-instance-bgp)# route-target import 1001:305
Router(config-evpn-instance-bgp)# route-target export 1001:5305
Router(config-evpn-instance-bgp)# exit
Router(config-evpn-instance)# etree
Router(config-evpn-instance-etree)# rt-leaf
Router(config-evpn-instance)# exit
Router(config-evpn-instance)# control-word-disable
Router(config-evpn-instance)# advertise-mac
Router(config-evpn-instance-mac)# commit

/* Configure bundle Ethernet */
Router# configure
Router(config)# interface Bundle-Ether700
Router(config-if)# lacp system mac 00aa.aabb.1010
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit

Router# configure
Router(config)# interface Bundle-Ether720
Router(config-if)# lacp system mac 00aa.aabb.1212
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit

/* Configure EVPN interface */
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether700
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.77.77.77.77.77.77.77.00
Router(config-evpn-ac-es)# bgp route-target 0000.0000.0001
Router(config-evpn-ac-es)# commit

Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether720
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.77.77.77.77.77.77.77.20
Router(config-evpn-ac-es)# bgp route-target 0000.0000.0020
Router(config-evpn-ac-es)# commit


/* Configure PE-04 (as leaf) */

/* Configure bridge domain */
Router # configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group EVPN_BD
Router(config-l2vpn-bg)# bridge-domain evpn_bvi_305
Router(config-l2vpn-bg-bd)# interface Bundle-Ether700.305
Router(config-l2vpn-bg-bd-ac)# split-horizon group
Router (config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# interface Bundle-Ether720.305
Router(config-l2vpn-bg-bd-ac)# split-horizon group
Router (config-l2vpn-bg-bd-ac)# exit
Router (config-l2vpn-bg-bd)# evi 305
Router (config-l2vpn-bg-bd-evi)# commit

/* Configure attachment circuit */
Router# configure
Router(config)# interface Bundle-Ether700.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

Router# configure
Router(config)# interface Bundle-Ether720.305 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 305
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

/* Configure EVPN EVI */
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 305
Router(config-evpn-instance)# bgp
Router(config-evpn-instance-bgp)# route-target import 1001:305
Router(config-evpn-instance-bgp)# route-target export 1001:5305
Router(config-evpn-instance-bgp)# exit
Router(config-evpn-instance)# etree
Router(config-evpn-instance-etree)# rt-leaf
Router(config-evpn-instance)# exit
Router(config-evpn-instance)# control-word-disable
Router(config-evpn-instance)# advertise-mac
Router(config-evpn-instance-mac)# commit

/* Configure bundle Ethernet */
Router# configure
Router(config)# interface Bundle-Ether700
Router(config-if)# lacp system mac 00aa.aabb.1010
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit

Router# configure
Router(config)# interface Bundle-Ether720
Router(config-if)# lacp system mac 00aa.aabb.1212
Router(config-if)# lacp switchover suppress-flaps 300
Router(config-if)# lacp cisco enable link-order signaled
Router(config-if)# bundle wait-while 100
Router(config-if)# commit

/* Configure EVPN interface */
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether700
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.77.77.77.77.77.77.77.00
Router(config-evpn-ac-es)# bgp route-target 0000.0000.0001
Router(config-evpn-ac-es)# commit

Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether720
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 00.77.77.77.77.77.77.77.20
Router(config-evpn-ac-es)# bgp route-target 0000.0000.0020
Router(config-evpn-ac-es)# commit
```

Running Configuration

This section shows the PE-01, PE-02, PE-3, and PE-04 running configuration.

```
/* PE-01 Configuration */
l2vpn
 bridge group EVPN_BD
  bridge-domain evpn_bvi_305
   interface Bundle-Ether800.305
   !
   interface Bundle-Ether820.305
   !
   evi 305
   !
  !
 !
interface Bundle-Ether800.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
```

```
!
interface Bundle-Ether820.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
!
evpn
 evi 305
  bgp
   route-target import 1001:305
   route-target export 1001:305
   route-target import 1001:5305
  !
  control-word-disable
  advertise-mac
  !
 !
!
interface Bundle-Ether800
 lacp system mac 00aa.aabb.2020
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
interface Bundle-Ether820
 lacp system mac 00aa.aabb.2222
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
evpn
 interface Bundle-Ether800
  ethernet-segment
   identifier type 0 00.88.88.88.88.88.88.88.00
   bgp route-target 0001.0000.0001
  !
 !
!
evpn
 interface Bundle-Ether820
  ethernet-segment
   identifier type 0 00.88.88.88.88.88.88.88.20
   bgp route-target 0001.0000.0020
  !
 !
!


/* PE-02 Configuration */
l2vpn
 bridge group EVPN_BD
  bridge-domain evpn_bvi_305
   interface Bundle-Ether800.305
   !
   interface Bundle-Ether820.305
   !
   evi 305
   !
  !
 interface Bundle-Ether800.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
!
interface Bundle-Ether820.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
```

```
!
evpn
 evi 305
  bgp
   route-target import 1001:305
   route-target export 1001:305
   route-target import 1001:5305
  !
  control-word-disable
  advertise-mac
  !
 !
interface Bundle-Ether800
 lacp system mac 00aa.aabb.2020
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
interface Bundle-Ether820
 lacp system mac 00aa.aabb.2222
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
evpn
 interface Bundle-Ether800
  ethernet-segment
   identifier type 0 00.88.88.88.88.88.88.88.00
   bgp route-target 0001.0000.0001
  !
 !
evpn
 interface Bundle-Ether820
  ethernet-segment
   identifier type 0 00.88.88.88.88.88.88.88.20
   bgp route-target 0001.0000.0020
  !
 !


/* PE-03 Configuration */
l2vpn
 bridge group EVPN_BD
  bridge-domain evpn_bvi_305
   interface Bundle-Ether700.305
    split-horizon group
   !
   interface Bundle-Ether720.305
    split-horizon group
   !
   evi 305
   !
  !
 !
interface Bundle-Ether700.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
!
interface Bundle-Ether720.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
!
evpn
 evi 305
  bgp
```

```
   route-target import 1001:305
   route-target export 1001:5305
  !
  etree
   rt-leaf
  !
  control-word-disable
  advertise-mac
  !
 !
!
interface Bundle-Ether700
 lacp system mac 00aa.aabb.1010
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
interface Bundle-Ether720
 lacp system mac 00aa.aabb.1212
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
evpn
 interface Bundle-Ether700
  ethernet-segment
   identifier type 0 00.77.77.77.77.77.77.77.00
   bgp route-target 0000.0000.0001
  !
 !
!
evpn
 interface Bundle-Ether720
  ethernet-segment
   identifier type 0 00.77.77.77.77.77.77.77.20
   bgp route-target 0000.0000.0020
  !
 !
!


/* PE-04 Configuration */
l2vpn
 bridge group EVPN_BD
  bridge-domain evpn_bvi_305
   interface Bundle-Ether700.305
    split-horizon group
   !
   interface Bundle-Ether720.305
    split-horizon group
   !
   evi 305
   !
  !
 !
interface Bundle-Ether700.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
!
interface Bundle-Ether720.305 l2transport
 encapsulation dot1q 305
 rewrite ingress tag pop 1 symmetric
!
evpn
 evi 305
```

```
  bgp
   route-target import 1001:305
   route-target export 1001:5305
  !
  etree
   rt-leaf
  !
  control-word-disable
  advertise-mac
  !
 !
!
interface Bundle-Ether700
 lacp system mac 00aa.aabb.1010
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
interface Bundle-Ether720
 lacp system mac 00aa.aabb.1212
 lacp switchover suppress-flaps 300
 lacp cisco enable link-order signaled
 bundle wait-while 100
!
evpn
 interface Bundle-Ether700
  ethernet-segment
   identifier type 0 00.77.77.77.77.77.77.77.00
   bgp route-target 0000.0000.0001
  !
 !
!
evpn
 interface Bundle-Ether720
  ethernet-segment
   identifier type 0 00.77.77.77.77.77.77.77.20
   bgp route-target 0000.0000.0020
  !
 !
!
```

Verification

This section shows how the L2 MAC addresses are synchronized as LOCAL and L2VPN with multihoming active-active peers PE. Also, the root PE is aware of MAC addresses learnt on leaf PE remotely through RT2 advertisements.

```
Router:PE-01# show l2route evpn mac all
Topo ID  Mac Address    Producer    Next Hop(s)
-------- -------------- ----------- ----------------------------------------
204      001f.0100.0001 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0001 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0002 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0002 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0003 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0003 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0004 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0004 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0005 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0005 L2VPN       Bundle-Ether820.305, N/A
204      0020.0100.0001 L2VPN       26791/I/ME, N/A
204      0020.0100.0002 L2VPN       26791/I/ME, N/A
```

```
204      0020.0100.0003 L2VPN       26791/I/ME, N/A
204      0020.0100.0004 L2VPN       26791/I/ME, N/A
204      0020.0100.0005 L2VPN       26791/I/ME, N/A
204      0021.0100.0001 L2VPN       Bundle-Ether800.305, N/A
204      0021.0100.0002 L2VPN       Bundle-Ether800.305, N/A
204      0021.0100.0003 LOCAL       Bundle-Ether800.305, N/A
204      0021.0100.0004 L2VPN       Bundle-Ether800.305, N/A
204      0021.0100.0005 LOCAL       Bundle-Ether800.305, N/A
204      0022.0100.0001 L2VPN       26790/I/ME, N/A
204      0022.0100.0002 L2VPN       26790/I/ME, N/A
204      0022.0100.0003 L2VPN       26790/I/ME, N/A
204      0022.0100.0004 L2VPN       26790/I/ME, N/A
204      0022.0100.0005 L2VPN       26790/I/ME, N/A

Router:PE-02# show l2route evpn mac all
Topo ID  Mac Address    Producer    Next Hop(s)
-------- -------------- ----------- ----------------------------------------
204      001f.0100.0001 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0001 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0002 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0002 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0003 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0003 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0004 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0004 L2VPN       Bundle-Ether820.305, N/A
204      001f.0100.0005 LOCAL       Bundle-Ether820.305, N/A
204      001f.0100.0005 L2VPN       Bundle-Ether820.305, N/A
204      0020.0100.0001 L2VPN       27367/I/ME, N/A
204      0020.0100.0002 L2VPN       27367/I/ME, N/A
204      0020.0100.0003 L2VPN       27367/I/ME, N/A
204      0020.0100.0004 L2VPN       27367/I/ME, N/A
204      0020.0100.0005 L2VPN       27367/I/ME, N/A
204      0021.0100.0001 LOCAL       Bundle-Ether800.305, N/A
204      0021.0100.0002 LOCAL       Bundle-Ether800.305, N/A
204      0021.0100.0003 L2VPN       Bundle-Ether800.305, N/A
204      0021.0100.0004 LOCAL       Bundle-Ether800.305, N/A
204      0021.0100.0005 L2VPN       Bundle-Ether800.305, N/A
204      0022.0100.0001 L2VPN       27366/I/ME, N/A
204      0022.0100.0002 L2VPN       27366/I/ME, N/A
204      0022.0100.0003 L2VPN       27366/I/ME, N/A
204      0022.0100.0004 L2VPN       27366/I/ME, N/A
204      0022.0100.0005 L2VPN       27366/I/ME, N/A
```

The following output shows how the multihoming PE is aware of its local L2 MAC addresses as well as the MAC addresses learnt on the root node only. Leaf multihoming PE is not aware of any other MAC addresses learnt on other leaf PE nodes except if they are learnt on a multihoming active-active ethernet-segment on the peer leaf PE.

```
Router:PE-03# show l2route evpn mac all
Topo ID  Mac Address    Producer    Next Hop(s)
-------- -------------- ----------- ----------------------------------------
200      0011.0100.0003 L2VPN       30579/I/ME, N/A
200      0011.0100.0005 L2VPN       30579/I/ME, N/A
204      001f.0100.0001 L2VPN       30588/I/ME, N/A
204      001f.0100.0002 L2VPN       30588/I/ME, N/A
204      001f.0100.0003 L2VPN       30588/I/ME, N/A
204      001f.0100.0004 L2VPN       30588/I/ME, N/A
204      001f.0100.0005 L2VPN       30588/I/ME, N/A
204      0020.0100.0001 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0001 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0002 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0002 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0003 LOCAL       Bundle-Ether720.305, N/A
```

```
204      0020.0100.0003 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0004 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0004 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0005 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0005 L2VPN       Bundle-Ether720.305, N/A
204      0021.0100.0001 L2VPN       30587/I/ME, N/A
204      0021.0100.0002 L2VPN       30587/I/ME, N/A
204      0021.0100.0003 L2VPN       30587/I/ME, N/A
204      0021.0100.0004 L2VPN       30587/I/ME, N/A
204      0021.0100.0005 L2VPN       30587/I/ME, N/A
204      0022.0100.0001 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0001 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0002 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0002 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0003 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0003 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0004 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0004 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0005 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0005 L2VPN       Bundle-Ether700.305, N/A

Router:PE-04# show l2route evpn mac all
Topo ID  Mac Address    Producer    Next Hop(s)
-------- -------------- ----------- ----------------------------------------
200      0011.0100.0003 L2VPN       30545/I/ME, N/A
200      0011.0100.0005 L2VPN       30545/I/ME, N/A
204      001f.0100.0001 L2VPN       30550/I/ME, N/A
204      001f.0100.0002 L2VPN       30550/I/ME, N/A
204      001f.0100.0003 L2VPN       30550/I/ME, N/A
204      001f.0100.0004 L2VPN       30550/I/ME, N/A
204      001f.0100.0005 L2VPN       30550/I/ME, N/A
204      0020.0100.0001 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0001 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0002 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0002 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0003 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0003 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0004 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0004 L2VPN       Bundle-Ether720.305, N/A
204      0020.0100.0005 LOCAL       Bundle-Ether720.305, N/A
204      0020.0100.0005 L2VPN       Bundle-Ether720.305, N/A
204      0021.0100.0001 L2VPN       30549/I/ME, N/A
204      0021.0100.0002 L2VPN       30549/I/ME, N/A
204      0021.0100.0003 L2VPN       30549/I/ME, N/A
204      0021.0100.0004 L2VPN       30549/I/ME, N/A
204      0021.0100.0005 L2VPN       30549/I/ME, N/A
204      0022.0100.0001 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0001 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0002 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0002 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0003 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0003 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0004 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0004 L2VPN       Bundle-Ether700.305, N/A
204      0022.0100.0005 LOCAL       Bundle-Ether700.305, N/A
204      0022.0100.0005 L2VPN       Bundle-Ether700.305, N/A
```

Related Topics

- EVPN E-Tree Using RT Constraints, on page 63

Associated Commands

- etree rt-leaf
- show l2route evpn mac all

EVPN E-Tree Per-PE (Scenario1b)

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
EVPN E-Tree Per-PE (Scenario1b)	Release 7.5.1	This feature allows you to configure an attachment circuit on a PE device either as a root site or a leaf site using the **etree leaf** label for a given bridge-domain. By preventing communication among leaf ACs connected to the same PE and belonging to the same bridge-domain, you can segregate traffic received and sent from different geographical locations. This segregation helps in securing traffic and Denial of Service Prevention (DoS). This feature is compliant with RFC 8317.

EVPN Ethernet Tree (E-Tree) is a rooted-multipoint Ethernet service over MPLS core and enables you to define attachment circuits (ACs) as either a root site or a leaf site. The provider edge (PE) nodes can receive L2 traffic either from the attachment circuit (AC) of a bridge domain (BD) or from the remote PE node. For a given BD, L2 communication can only happen from root to leaf and leaf to root, and root to root. L2 communication between the ACs of two or more leafs is not allowed.

You can implement E-Tree in the following two ways:

- Scenario 1 - All ACs at a particular PE for a given BD can be either root or leaf site and all traffic for an EVI from a PE in the network is from either a root or a leaf. In this scenario you have two options to configure E-Tree:
  - Scenario 1a - You can configure E-Tree with route-targets (RT) constraints using two RTs per EVI. For more information, see the *EVPN E-Tree Using RT Constraints* section.
  - Scenario 1b - You can configure E-Tree without route-targets (RT) constraints and using **etree leaf** label.
- Scenario 2 - A PE for a given EVI can have both root and leaf sites. The root and leaf designation increases from bridge-domain level in Scenario 1, to a per AC level. For more information, see the *EVPN ETREE Per-AC (Scenario 2)* section.

Scenario 1b

In this scenario, you can configure E-Tree without route-targets (RT) constraints and using **etree leaf** label.

When you configure E-Tree with Scenario 1b, for known unicast traffic, MAC advertisements originating from a leaf site is identified with an **etree leaf** label to classify that the source is a leaf. Ingress filtering is performed, and traffic originating at leaf AC destined for a remote leaf MAC is dropped. If the remote PE is also a leaf, the ingress traffic from the source leaf is dropped. If the remote PE is a root, the ingress traffic from the source leaf is forwarded.

For BUM traffic, egress filtering is performed and leaf nodes transmit an **etree leaf** label to identify that leaf sites are connected to the PE. Then, at the ingress node, BUM traffic originating from a leaf node is tagged with the corresponding remote **etree leaf** label. At the egress PE, traffic is tagged with the matching **etree leaf** label that is dropped at leaf ACs.

Scenario 1b Behavior

- E-Tree leaf is configured per bridge domain. If there is no leaf configuration, the bridge domain is always a root.
- All ACs inherit E-Tree leaf designation from the bridge domain.
- Split-horizon group between ACs of a leaf is enabled automatically.
- All local MACs learned under the BD is advertised to BGP with **etree leaf** indicator.
- Upon first leaf configuration, a special E-Tree ethernet segment with ESI-0 is created to allocate a split-horizon label, referred to as the local etree leaf label.
- ES/EAD with ESI-0 (ES-0/EAD) is advertised to BGP with etree leaf label.

You can use scenario 1b to interact with scenario 2 when a remote node is NCS 5500, NCS 540, or NCS 560, and configure E-Tree using scenario 1b with ASR 9000.

Configure EVPN E-Tree Per-PE (Scenario1b)

Perform this task to configure EVPN E-Tree Per-PE (Scenario1b).

Configure EVPN E-Tree leaf per bridge domain.

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd_1
Router(config-l2vpn-bg-bd)# etree
Router(config-l2vpn-bg-bd-etree)# leaf
Router(config-l2vpn-bg-bd-etree)# interface Bundle-Ether400.1
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd-etree)#  interface Bundle-Ether401.1001
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd-etree)# interface Bundle-Ether4701.2001
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# routed interface BVI1
Router(config-l2vpn-bg-bd-bvi)# split-horizon group core
Router(config-l2vpn-bg-bd-bvi)# exit
Router(config-l2vpn-bg-bd)# evi 200
Router(config-l2vpn-bg-bd-evi)# commit
```

Running Configuration

```
l2vpn
 bridge group bg1
  bridge-domain bd_1
   etree
```

```
     leaf
    !
    interface Bundle-Ether400.1
    !
    interface Bundle-Ether401.1001
    !
    interface Bundle-Ether4701.2001
    !
    routed interface BVI1
     split-horizon group core
    !
    evi 200
    !
   !
  !
```

EVPN ETREE Per-AC (Scenario 2)

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
EVPN E-TREE Per-AC (Scenario 2)	Release 7.5.1	This feature allows a provider edge (PE) device to have both root and leaf sites for a given EVI. This feature increases the granularity of leaf designation from the entire bridge to AC bridge ports; ACs under a bridge may be root or leaf.

Customer sites represented by ACs can be defined as root or leaf. PE can receive traffic from both root and leaf ACs for a given EVI from a remote node. An EVI can be associated with both root(s) and leaf(s). If an AC is not configured as E-Tree leaf, it is considered as root by default.

In Scenario 2, a PE for a given EVI can have both root and leaf sites. The granularity of root or leaf designation increases from bridge-domain level in Scenario 1 to a per AC level. Traffic for an EVI from a PE in the network can be from either a root or a leaf site.

Scenario 2 Behavior

For unicast traffic:

- Remote PE performs ingress filtering to prevent traffic from being sent unnecessarily over the core to be filtered at egress PE.
- PE indicates if the MAC address is associated with a root or a leaf.
- MAC advertisements originating from a leaf site are colored with a leaf-indication flag in an extended community; routes that do not have this flag are from a root site.
- Remote PEs perform ingress filtering, when MAC is programmed and the leaf-indication is present, a cross-check is performed with the originating AC; if the AC is also a leaf, packets will not be forwarded.
- Supports E-Tree extcomm of type 0x06 (EVPN) and sub-type 0x05 used for leaf-indication for known-unicast and BUM traffic.
- Enable unknown unicast suppression at EVIs connected to both root and leaf sites to prevent egress unknown unicast traffic arriving at an EVI from being flooded to ACs. This eliminates the leaf-to-leaf traffic during a BD MAC flush.

- MAC advertises local ESI and there is no leaf indicator from root.

  When processing the root sync-route, the root or leaf status of the individual AC is considered, instead of the entire bridge-domain. If a root MAC with a matching local ESI is received, and if the corresponding AC is a leaf, a syslog message is generated for the misconfiguration.

For BUM traffic:

- The PE performs egress filtering for BUM traffic. BUM traffic originating from leaf sites is filtered at egress nodes if the destination is also a leaf.
- A PE with leaf sites allocate a leaf label, and communicate this label to remote PEs using an ES/EAD route with ESI 0 with the ETREE extended community.
- BUM traffic originating from single-homed leaf AC is tagged with destination **etree leaf** label.
- BUM traffic originating from single-homed root AC is not tagged with any ESI or **etree leaf** label.
- BUM traffic originating from multi-homed leaf AC is tagged with destination **etree leaf** label.
- BUM traffic originating from multi-homed root AC is tagged with ESI label.
- The ingress PE tags MPLS frames with the **etree leaf** label for traffic originating from a leaf site; this label allows the disposition PE to perform egress filtering to native EVPN ESI label filtering.
- Intra-PE forwarding between leaf sites is prevented by putting all leaf ACs under a given bridge domain in a single split-horizon group.

Restrictions

- This feature is supporeted only on physical and bundle interfaces.
- You can use either Scenario 1 or Scenario 2 for a given EVI within a network.
- Scenario 2 interoperates only with scenario 1b and vice-versa.
- You can use scenario 1b to interact with scenario 2 when a remote node is NCS 5500, NCS 540, or NCS 560, and configure E-Tree using scenario 1b with ASR 9000.