Configure EVPN IRB, Distributed Anycast Gateway and E-tree

This chapter introduces you to Ethernet VPN (EVPN) Integrated Routing and Bridging (IRB), Distributed Anycast Gateway, and E-Tree features and their description.

- EVPN IRB , on page 1
- EVPN Single-Homing Access Gateway , on page 3
- EVPN Multihoming All-Active, on page 4
- EVPN Single-Active Multihoming for Anycast Gateway IRB, on page 5
- Enable Auto-BGP RT with Manual ESI Configuration, on page 9
- Supported EVPN IRB Scenarios, on page 9
- Distributed Anycast Gateway, on page 9
- BVI-Coupled Mode, on page 12
- EVPN-IRB ARP and ND proxy suppression, on page 14
- VM Mobility Support, on page 24
- Configure EVPN IRB, on page 26
- EVPN IPv6 Hosts with Mobility, on page 43
- Duplicate IP Address Detection, on page 58
- EVPN Automatic Unfreezing of MAC and IP Addresses, on page 60
- EVPN E-Tree, on page 62
- EVPN E-Tree Using RT Constraints, on page 70
- EVPN E-Tree Per-PE (Scenario 1b), on page 85
- DHCPv4 Relay on IRB, on page 91
- DHCPv4 Relay Synchronization for All-Active Multihoming, on page 98
- DHCPv6 Relay IAPD on IRB, on page 99
- DHCPv6 PD Synchronization for All-Active Multihoming using Session Redundancy , on page 102
- IAPD Route Distribution and Withdrawal in DHCPv6 Relay, on page 105
- Layer 3 EVPN IGMP and MLD state synchronization, on page 105

EVPN IRB

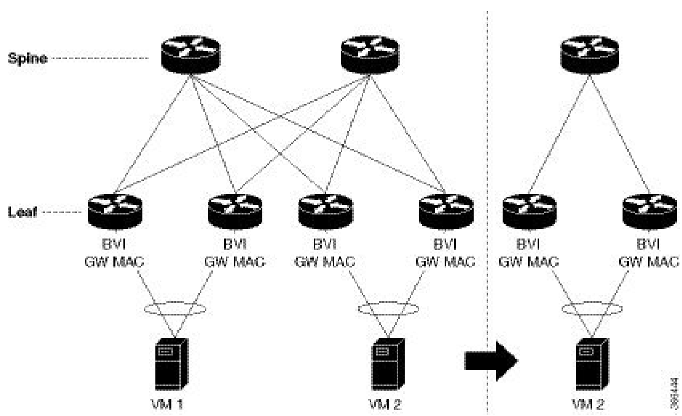
EVPN IRB feature enables a Layer 2 VPN and an Layer 3 VPN overlay that allows end hosts across the overlay to communicate with each other within the same subnet and across different subnets within the VPN.

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.

Figure 1: EVPN IRB



The benefit of EVPN IRB is that it allows the hosts in an IP subnet to be provisioned anywhere in the data center. When a virtual machine (VM) in a subnet is provisioned behind a EVPN PE, and another VM is required in the same subnet, it can be provisioned behind another EVPN PE. The VMs do not have to be localized; they need not be directly connected; or be in the same complex. The VM is allowed to move across in the same subnet. Availability of IP MPLS network across all the EVPN PEs enables the provisioning of VM mobility. The EVPN PEs route traffic to each other through MPLS encapsulation.

The EVPN PEs are connected to each other by a spine so they have IP reachability to each other's loopback interfaces. The IP network and MPLS tunnels existing between these EVPN PEs constitute the IP MPLS underlay fabric.

You can configure the MPLS tunnels to tunnel Layer 2 traffic, and to overlay VPN on these tunnels. EVPN control plane distributes both Layer 2 MAC reachability and Layer 3 IP reachability for hosts within the context of the VPN; it overlays a tenant's VPN network on top of the MPLS underlay fabric. Thus you can have tenant's hosts, which are in the same subnet layer 2 domain, but distributed across the fabric, communicate to each other as if they are in a Layer 2 network.

The Layer 2 VLAN and the corresponding IP subnet are not only a network of physically connected hosts on Layer 2 links, but an overlayed network on top of underlayed IP MPLS fabric which is spread across the datacenter.

A routing service, which enables stretching of the subnet across the fabric, is available. It also provides Layer 3 VPN and performs routing between subnets within the context of the Layer 3 VPN. The EVPN PEs provide Layer 2 bridging service between hosts that are spread across the fabric within a Layer 2 domain that is stretched across the fabric, and Layer 3 VPN service or inter-subnet routing service for hosts in different subnets within Layer 3 VPN. For example, as shown in the above topology diagram, the two VM are in the same subnet but they are not connected directly through each other through a Layer 2 link. The Layer 2 link is replaced by MPLS tunnels that are connecting them. The whole fabric acts as a single switch and bridges traffic from one VM to the other. This also enables VM mobility.

Note Egress marking is not supported on L2 interfaces in a bridge domain.

In the above topology diagram, the VMs, VM1 and VM2 are connected each other. When VM2 migrates to a different switch and different server, the VM's current MAC address and IP address are retained. When the subnet is stretched between two EVPN PEs, the same IRB configuration is applied on both the devices.

For stretching within the same subnet, you must configure the AC interface and the EVI; it is not required to configure IRB interface or VRF.

Note Only a single custom MAC address is supported for all BVIs across the system.

Limitations

In case static MAC address is configured on a bundle-ether interface, the following limitations are applied:

- Locally generated packets, such as ICMP, BGP, and so on, going out from the interface have the source MAC address as the statically configured MAC address.
- Transit (forwarded) packets going out of the interface do not have the configured static MAC as source MAC address. In such a scenario, the upper 36-bits come from the system MAC address (or the original/dynamic MAC address) and the lower 12-bits set as zero. To check the dynamic pool of MAC addresses included, use the show ethernet mac-allocation detail command.

  For example, if the dynamic MAC address is 008A.9624.48D8 and the configured static MAC address is 0011.2222.1111. Then, the source MAC for transit (forwarded) traffic will be 008A.9624.4000.
- To prevent recursive lookup on the egress PE, avoid filtering host routes from EVPN VPNv4 redistribution; instead, inject host routes into the VPNv4 domain.

For more information on limitations, refer *Limitations and Compatible Characteristics of Ethernet Link Bundles* in *Interface and Hardware Component Configuration Guide for Cisco NCS 540 Series Routers*

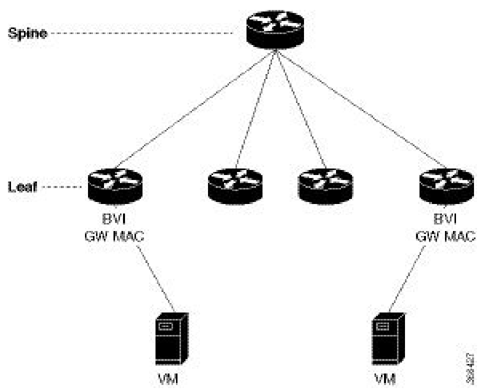
EVPN Single-Homing Access Gateway

The EVPN provider edge (PE) devices learn the MAC address and IP address from the ARP traffic that they receive from the customer edge (CE) devices. The PEs create the MAC+IP routes. The PEs advertise the MAC+IP routes to MPLS core. They inject the host IP routes to IP-VPN gateway. Subnet routes are also advertised from the access EVPN PEs in addition to host routes. All the PE nodes add the host routes in the IP-VRF table. The EVPN PE nodes add MAC route to the MAC-VRF table. The IP-VPN PE advertise the subnet routes to the provider edge devices which add the subnet routes to IP-VRF table. On the PE devices, IRB gateway IP addresses and MAC addresses are not advertised through BGP. IRB gateway IP addresses or MAC addresses are used to send ARP requests towards the datacenter CEs.

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.

Figure 2: EVPN Single-Homing Access Gateway



The above topology depicts how EVPN single-homing access gateway enables network connectivity by allowing a CE device to connect to one PE device. The PE device is attached to the Ethernet Segment through bundle or physical interfaces. Null Ethernet Segment Identifier (ESI) is used for single-homing.

EVPN Multihoming All-Active

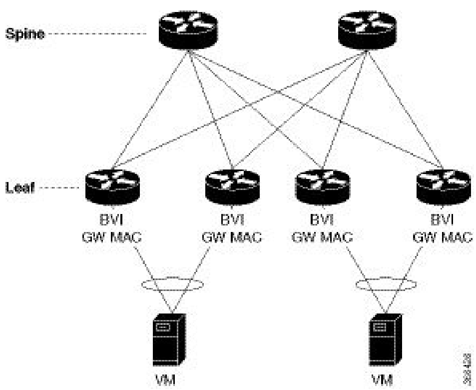
In EVPN IRB, both EVPN and IP VPN (both VPNv4 and VPNv6) address families are enabled between routers and Data Center Interconnect (DCI) gateways. When Layer 2 (L2) stretch is not available in multiple data centers (DC), routing is established through VPNv4 or VPNv6 routes. When Layer 2 stretch is available, host routing is applied where IP-MAC routes are learnt by ARP 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.

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.

Figure 3: EVPN Multi-Homing All-Active



The above topology describes how EVPN Multi-homing access gateway enables redundant network connectivity by allowing a CE device to connect to more than one PE device. 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. The All-Active Link Aggregation Group bundle operates as an Ethernet segment. Only MC bundles that operates between two chassis are supported.

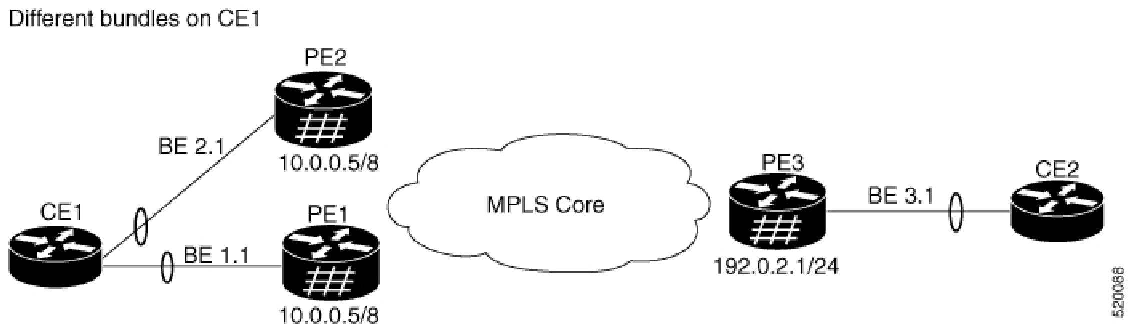
EVPN Single-Active Multihoming for Anycast Gateway IRB

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	Release 7.3.1	This feature is now supported on Cisco NCS 5700 series fixed port routers and the Cisco NCS 5500 series routers that have the Cisco NC57 line cards installed and operating in the native and compatible modes.

The EVPN Single-Active Multihoming for Anycast Gateway IRB feature supports single-active redundancy mode. In this mode, the provider edge (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.

Figure 4: EVPN: Single-Active Multihoming for Anycast Gateway IRB



Consider a topology where CE1 is multihomed to PE1 and PE2. Bundle Ethernet interfaces BE 1.1, BE 2.1, and the ingress interface must belong to the same switching domain on CE1. Enable host routing and configure anycast gateway IP address on both these peering PEs. PE1 and PE2 are connected to PE3 through MPLS core. PE3 has reachability of subnet 10.0.0.5/8 to both peering PEs. Peering PEs has reachability to PE3 subnet 192.0.2.1/24. CE2 is connected to PE3 through an Ethernet interface bundle. PE1 and PE2 advertise Type 4 routes, and then performs designated forwarder (DF) election. The non-DF blocks the traffic in both the directions in single-active mode.

Consider a traffic flow from CE1 to CE2. CE1 sends an address resolution protocol (ARP) broadcast request to both PE1 and PE2. Peering PEs performs designated forwarder (DF) election for shared ESI. If PE1 is the designated forwarder for the EVI, PE1 replies to the ARP request from CE1. PE2 drops the traffic from CE1. Thereafter, all the unicast traffic is sent through PE1. PE2 is set to stand-by or blocked state and traffic is not sent over this path. PE1 advertises MAC to PE3. PE3 always sends and receives traffic through PE1. PE3 sends the traffic to CE2 over Ethernet interface bundle. If BE1 fails, PE2 becomes the DF and traffic flows through PE2.

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
```

Enable Auto-BGP RT with Manual ESI Configuration

Configuring an ES-Import RT was previously mandatory for Type 0 ESI. The ES-Import RT is auto-extracted by default, and the configuration serves to override the default value. This feature is based on RFC 7432 but applied specifically to ESI Type 0. For more information, see Section 5 of RFC 7432.

Supported EVPN IRB Scenarios

EVPN IRB supports the following scenarios:

Dual-homing supports the following methods:

- Only all-active mode is supported
- Only two PE gateways in a redundancy group

Single-homing supports the following methods:

- Physical
- VLAN
- Bundle-ethernet
- QinQ access
- Only IPv4 is supported.
- Subnet-stretch feature with EVPN IRB is supported in VRF as well as in Global Routing Table (GRT). In GRT, **bgp implicit-import** under the BGP address-family l2vpn evpn must be configured.

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 are the solutions that are part of the Distributed Anycast Gateway feature:

EVPN IRB with All-Active Multi-Homing without Subnet Stretch or Host-Routing across the Fabric

For those subnets that are local to a set of multi-homing EVPN PEs, EVPN IRB Distributed Anycast Gateway is established through subnet routes that are advertised using EVPN Route Type 5 to VRF-hosting remote leafs. Though there is no need for the /32 routes within the subnet to be advertised, host MAC and ARP entries have to synced across the EVPN PE to which the servers are multi-homed.

Note The Subnet Stretch feature with EVPN IRB is exclusively available for use within VRF instances and is not applicable to the global VRF.

This type of multi-homing has the following characteristics:

- All-active EV LAG on access
- Layer 3 ECMP for the fabric for dual-homed hosts based on subnet routes
- Absence of Layer 2 subnet stretch over the fabric
- Layer 2 stretch within redundancy group of leafs with orphan ports

Prefix-routing solution for a non-stretched subnet is summarized as below:

Across multi-homing EVPN PEs:

- Local ARP cache and MAC addresses are synchronized for dual-homed hosts through EVPN MAC+IP host route advertisements. They are imported as local, and are based on the local ESI match, for optimal forwarding to the access gateway.
- Orphan MAC addresses and host IP addresses are installed as remote addresses over the fabric.
- ES/EAD routes are exchanges for the designated forwarder (DF) election and split-horizon label.

Across remote EVPN PEs:

- Dual-homed MAC+IP EVPN Route Type 2 is exchanged with the ESI, EVI Label, Layer 2-Route Type. It is not imported across the fabric, if there is no subnet stretch or host-routing.
- The subnet IP EVPN Route Type 5 is exchanged with VRF label and Layer 3-Route Type.
- Layer 3 Route Type for the VRFs is imported that are present locally.
- Layer 2 Route Type for locally present BDs is imported. It is only imported from the leaf in the same redundancy group, if BD is not stretched.

EVPN IRB with All-Active Multihoming 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 multihoming has the following characteristics:

- All-active EV-LAG on the access gateway
- 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 host routing solution for a stretched subnet is summarized as follows:

Across multihoming EVPN PEs:

- The Local ARP cache and MAC addresses are synchronized for dual-homed hosts through EVPN MAC+IP host route advertisements. They are imported as local, based on the local ESI match, for optimal forwarding to the access gateway.
- Synchronized MAC+IP are re-originated for inter-subnet Layer 3 ECMP.
- Orphan MAC address and host IP address are installed as remote addresses over the fabric.
- ES/EAD route is exchanged for designated forwarder (DF) election and split-horizon label.

Across remote EVPN PEs:

- Dual-homed MAC+IP EVPN Route Type 2 is exchange with ESI, EVI label, Layer 2-Route Type, VRF label, and Layer 3-Route Type.
- Subnet IP EVPN Route Type 5 is exchanged for VRF label, Layer 3-Route Type for silent hosts, and non-stretched subnets.
- Layer 3 Route Type is imported for locally present VRFs.
- Layer 2 Route Type is imported for locally present bridge domains.

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.

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.

Note If custom or static MAC address is configured on a BVI interface, the MAC address on the wire may be different than what is configured. This has no operational or functional impact.

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

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
---------------------- --------------------
```

EVPN-IRB ARP and ND proxy suppression

The EVPN-IRB ARP and ND suppression is a network feature that:

- manages and minimizes ARP and ND traffic on a network segment,
- reduces bandwidth usage by suppressing unnecessary broadcast traffic, and
- provides flexibility to configure modes as specified in RFC 9161, Operational Aspects of Proxy ARP/ND in Ethernet Virtual Private Networks.

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
EVPN-IRB ND partial suppression mode	Release 25.1.1	You can now manage and minimize Neighbor Discovery (ND) for IPv6 traffic on a network segment by configuring partial suppression mode. You can configure EVPN-IRB ARP and ND proxy full or partial suppression modes for these features: • EVPN-IRB multi-homing • VM Mobility

EVPN-IRB ARP and ND proxy suppression	Release 24.4.1	You can now manage and minimize Address Resolution Protocol (ARP) for IPv4 traffic or Neighbor Discovery (ND) for IPv6 traffic on a network segment by configuring full or partial suppression mode. If the target host entry is not found in the ARP and ND tables, • the full suppression mode prevents ARP and ND request flooding and reduces the consumption of processing resources on network devices. • the partial suppression mode broadcasts the packet to all devices within the EVPN Bridge Domain (BD) as it would perform without suppression, ensuring connectivity even if the information isn't immediately available in the EVPN control plane. The feature introduces these changes: **CLI:** • **arp evpn-proxy mode** • **ipv6 nd evpn-proxy mode** **YANG Data Model:** • `Cisco-IOS-XR-um-ipv6-nd-cfg.yang` • `Cisco-IOS-XR-um-if-arp-cfg.yang` (see GitHub, YANG Data Models Navigator)

ARP and ND request flooding

When a device does not have the target entry such as IP-to-MAC mapping in its ARP and ND tables, the device sends an ARP and ND request to all devices in the network. This flooding can cause network congestion and performance issues, especially in environments with many devices.

ARP and ND suppression modes

You can manage and minimize ARP and ND traffic on a network segment by using full suppression or partial suppression modes.

Release 24.4.1 does not support ND partial suppression mode.

When you enable...	And...	Then...
ARP and ND full suppression mode on a PE router	if the target host entry is found in the ARP and ND tables,	the local PE router generates and send an ARP and ND reply using the remote IP and MAC address information from the EVPN control plane.

When you enable...	And...	Then...
ARP and ND full suppression mode on a PE router	if the target host entry is not found in the ARP and ND tables,	the router drops the ARP and ND request and the local PE does not send any replies to the host. This prevents the ARP and ND requests from flooding the network, which in turn reduces network congestion and saves processing resources on the devices.
ARP partial suppression mode on a PE router	if the target host entry is found in the ARP,	the local PE router generates and send an ARP reply using the remote IP and MAC address information from the EVPN control plane.
ARP partial suppression mode on a PE router	if the target host entry is not found in the ARP,	the network broadcasts the packet to all devices within the EVPN BD. This approach allows the network to handle the request as it would without suppression, ensuring connectivity even if the information isn't immediately available in the EVPN control plane.

From Release 25.1.1, You can manage and minimize ND traffic on a network segment by using partial suppression mode.

When you enable...	And...	Then...
ND partial suppression mode on a PE router	if the target host entry is found in the ND tables,	the local PE router generates and send an ND reply using the remote IP and MAC address information from the EVPN control plane.
ND partial suppression mode on a PE router	if the target host entry is not found in the ND tables,	the network broadcasts the packet to all devices within the EVPN BD. This approach allows the network to handle the request as it would without suppression, ensuring connectivity even if the information isn't immediately available in the EVPN control plane.

This feature supports:

- BVI interfaces.
- Duplicate IP address detection feature, which automatically detects any host with a duplicate IP address and blocks all MAC-IP routes that have a duplicate IP address, as described in Duplicate IP Address Detection.
- EVPN-IRB multi-homing networks from Release 25.1.1.

- VM Mobility from Release 25.1.1.
- EVPN-IRB ND partial suppression mode from Release 25.1.1.

Limitations for EVPN-IRB ARP and ND proxy suppression

This feature does not support:

- EVPN-IRB multi-homing networks in Release 24.4.1.
- EVPN-IRB ND partial suppression mode in Release 24.4.1.
- Anycast on bridge domains or interfaces.
- Static ARP on layer 3 interfaces.

These are the limitations:

- During mobility, confirmation requests are sent immediately to verify the existence of the host locally.
- If both the requesting CE device and the target CE device are connected to the same PE device through the same Attachment Circuit (AC) then the local PE will not send any ARP and ND reply to the host.

How EVPN-IRB ARP and ND suppression process works

Summary

These are the key components involved in EVPN-IRB ARP and ND suppression process:

- **Customer edge routers**: CE1, CE2, CE3, and CE4
- **Provider edge routers**: PE1, PE2, and PE3
- CE3 sends an ARP request for CE4 through PE3, and
- the target CE4 does not have an entry in the ARP and ND tables of PE3.

Workflow

Figure 5: EVPN-IRB Network without ARP and ND Proxy Full Suppression or with ARP Proxy Partial Suppression

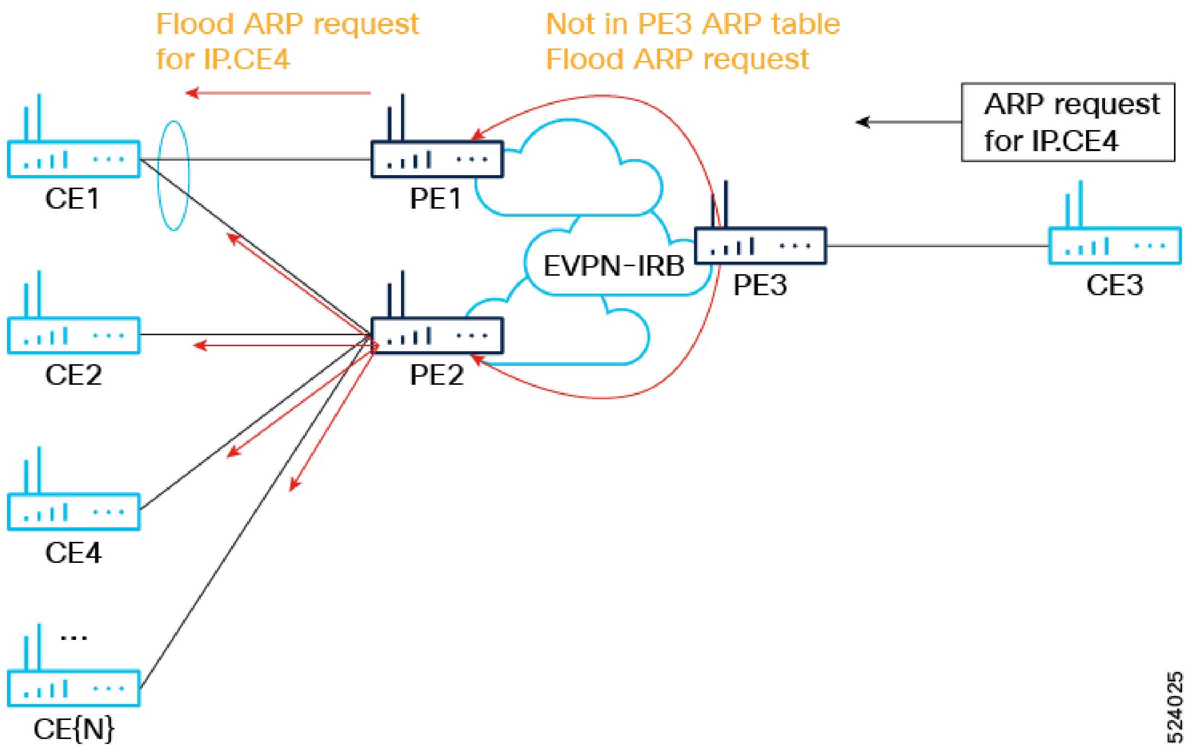
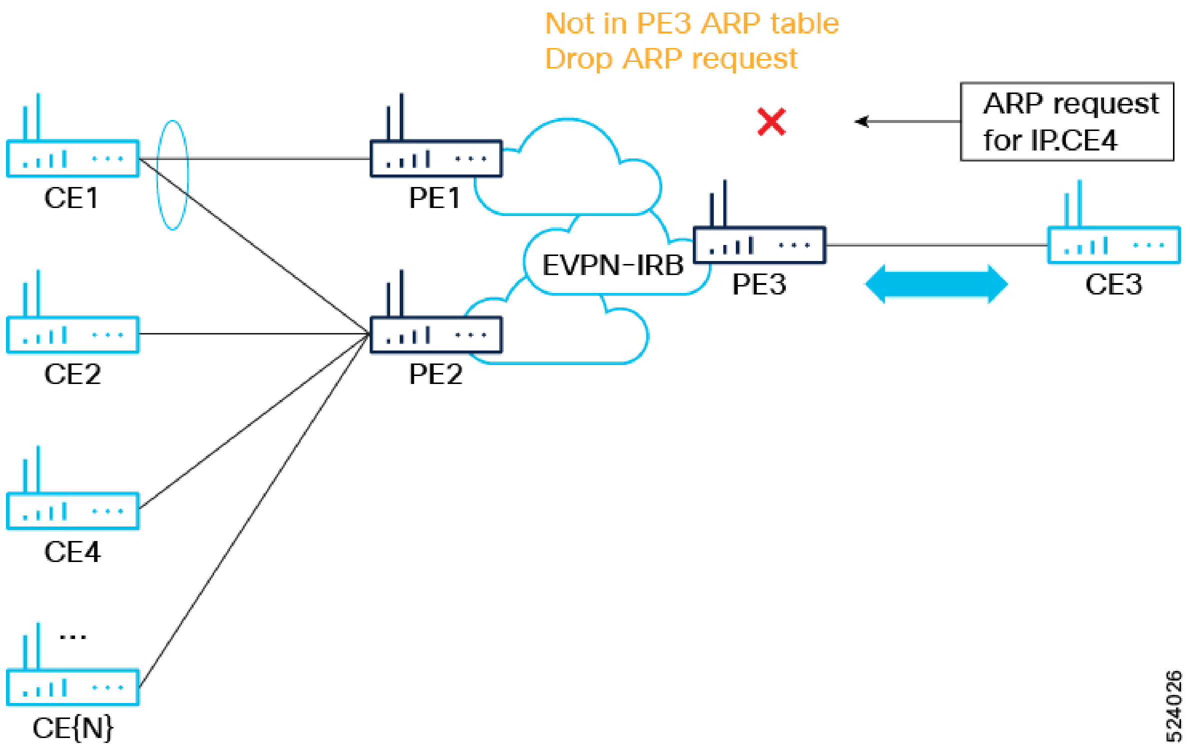


Figure 6: EVPN-IRB Network with ARP and ND Proxy Full Suppression



These stages describe how EVPN-IRB ARP and ND suppression process works.

When...	Then...
ARP and ND suppression mode is not configured or ARP partial suppression mode is configured on the EVPN-IRB network	ARP and ND requests are sent to all CE devices, from CE1 to CE{N}.
ARP and ND full suppression is enabled and if the target entry is not found in the ARP and ND tables of PE3,	the request is dropped by PE3, preventing network flooding.

From Release 25.1.1, You can now manage and minimize ND traffic on a network segment by using partial suppression mode.

When...	Then...
ND partial suppression mode is configured on the EVPN-IRB network	ND requests are sent to all CE devices, from CE1 to CE{N}.

You can configure these modes on PE3:

- ARP full suppression mode using the **arp evpn-proxy mode full-suppression** command, see Configure EVPN-IRB ARP proxy full or partial suppression and ND full or partial suppression, on page 20.

- ARP partial suppression mode using the **arp evpn-proxy mode partial-suppression** command, see Configure EVPN-IRB ARP proxy full or partial suppression and ND full or partial suppression, on page 20.
- ND full suppression mode using the **ipv6 nd evpn-proxy mode full-suppression** command, see Configure EVPN-IRB ARP proxy full or partial suppression and ND full or partial suppression, on page 20.
- From Release 25.1.1, ND partial suppression mode using the **ipv6 nd evpn-proxy mode partial-suppression** command, see Configure EVPN-IRB ARP proxy full or partial suppression and ND full or partial suppression, on page 20.
- (Optional) ARP full and partial suppression mode without BVI IPv4 address using the **arp evpn-proxy mode full-suppression** and **arp evpn-proxy mode partial-suppression** command, see Configure EVPN-IRB ARP proxy suppression without BVI IPv4 address, on page 22.
- (Optional) ND full suppression mode without BVI IPv6 address using the **ipv6 nd evpn-proxy mode full-suppression** command, see Configure EVPN-IRB ND proxy full or partial suppression without BVI IPv6 address, on page 23.
- (Optional) From Release 25.1.1, ND partial suppression mode without BVI IPv6 address using the **ipv6 nd evpn-proxy mode partial-suppression** command, see Configure EVPN-IRB ND proxy full or partial suppression without BVI IPv6 address, on page 23.

Configure EVPN-IRB ARP proxy full or partial suppression and ND full or partial suppression

Perform these steps to configure the EVPN-IRB ARP full or partial proxy suppression and ND full or partial suppression:

Before you begin

Configure VRFs and Route Targets (RT), see Configure EVPN IRB, on page 26.

Procedure

Step 1 Perform these steps to configure ARP and ND proxy suppression in the PE router:

- Configure the ARP and ND full suppression mode on BVI interface in the PE router.

Example:

```
Router# configure
Router(config)# interface BVI1
Router(config-if)# ipv4 address 10.0.1.3 255.255.255.0
Router(config-if)# ipv6 address 2001:DB8::1/32
Router(config-if)# host-routing
Router(config-if)# vrf foo
Router(config-if)# arp evpn-proxy mode full-suppression
Router(config-if)# ipv6 nd evpn-proxy mode full-suppression
Router(config-if)# exit
Router(config)# commit
```

- Configure the ARP partial suppression mode on BVI interface in the PE router.

Example:

```
Router# configure
Router(config)# interface BVI1
Router(config-if)# ipv4 address 10.0.1.3 255.255.255.0
Router(config-if)# host-routing
Router(config-if)# vrf foo
Router(config-if)# arp evpn-proxy mode partial-suppression
Router(config-if)# exit
Router(config)# commit
```

- From Release 25.1.1, configure the ND partial suppression mode on BVI interface in the PE router.

Example:

```
Router# configure
Router(config)# interface BVI1
Router(config)# ipv6 enable
Router(config-if)# ipv6 address 2001:DB8::1/32
Router(config-if)# host-routing
Router(config-if)# vrf foo
Router(config-if)# ipv6 nd evpn-proxy mode partial-suppression
Router(config-if)# exit
Router(config)# commit
```

Step 2 Configure the L2VPN bridge domain and associate the BVI with the EVI.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd1
Router(config-l2vpn-bg-bd)# interface HundredGigE0/0/0/0
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# routed interface BV1
Router(config-l2vpn-bg-bd)# evi 1
Router(config-l2vpn-bg-bd)# exit
Router(config-l2vpn-bg)# exit
Router(config-l2vpn)# exit
Router(config)# commit
```

Step 3 Verify the configuration with the **show running-config** command.

Example:

```
Router# sh running-config interface BVI1
interface BVI1
host-routing
vrf vrf1
ipv4 address 10.0.1.3 255.255.255.0
arp evpn-proxy mode full-suppression
ipv6 nd evpn-proxy mode full-suppression
ipv6 address 2001:DB8::1/32
ipv6 enable
mac-address 0011.1111.1111
!
```

Configure EVPN-IRB ARP proxy suppression without BVI IPv4 address

Perform these steps to configure the EVPN-IRB ARP proxy full or partial suppression with BVI unnumbered interface and without BVI IP address:

Before you begin

Configure VRFs and RT, see Configure EVPN IRB, on page 26.

Procedure

Step 1 Configure loopback interface on PE and assign IPv4 address.

Example:

```
Router# configure
Router(config)# interface Loopback0
Router(config-if)# vrf vrf1
Router(config-if)# ipv4 address 192.168.0.1 255.255.255.255
```

Step 2 Perform these steps to configure the EVPN-IRB ARP proxy full or partial suppression mode without BVI IPv4 address on BVI interface in the PE router:

- Configure the ARP full suppression mode on BVI interface in the PE router.

  Example:

```
Router# configure
Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# vrf vrf1
Router(config-if)# ipv4 point-to-point
Router(config-if)# ipv4 unnumbered Loopback0
Router(config-if)# arp evpn-proxy mode full-suppression
Router(config-if)# exit
Router(config)# commit
```

- Configure the ARP partial suppression mode on BVI interface in the PE router.

  Example:

```
Router# configure
Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# vrf vrf1
Router(config-if)# ipv4 point-to-point
Router(config-if)# ipv4 unnumbered Loopback0
Router(config-if)# arp evpn-proxy mode partial-suppression
Router(config-if)# exit
Router(config)# commit
```

Step 3 Configure the L2VPN bridge domain and associate the BVI with the EVI.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd1
Router(config-l2vpn-bg-bd)# interface HundredGigE0/0/0/0
```

```
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# routed interface BV1
Router(config-l2vpn-bg-bd)# evi 1
Router(config-l2vpn-bg-bd)# exit
Router(config-l2vpn-bg)# exit
Router(config-l2vpn)# exit
Router(config)# commit
```

Step 4 Verify the configuration with the **show running-config** command.

Example:

```
Router# sh running-config interface BVI1
interface BVI1
host-routing
ipv4 point-to-point
ipv4 unnumbered Loopback0
arp evpn-proxy mode full-suppression
mac-address 0011.1111.1111
```

```
Router# sh running-config interface Loopback0
interface Loopback0
ipv4 address 192.168.0.1 255.255.255.255
!
```

Configure EVPN-IRB ND proxy full or partial suppression without BVI IPv6 address

Perform these steps to configure the EVPN-IRB ND proxy full or partial suppression without BVI IP address:

Before you begin

Configure VRFs and RT, see Configure EVPN IRB, on page 26.

Procedure

Step 1 Perform these steps to configure the EVPN-IRB ND proxy suppression mode without BVI IPv6 address on BVI interface in the PE router.

- Configure the ND proxy full suppression mode on BVI interface in the PE router.

  Example:

  ```
  Router# configure
  Router(config)# interface BVI1
  Router(config-if)# host-routing
  Router(config-if)# vrf vrf1
  Router(config-if)# ipv6 nd evpn-proxy mode full-suppression
  Router(config-if)# ipv6 enable
  Router(config-if)# exit
  Router(config)# commit
  ```

- From Release 25.1.1, configure the ND proxy partial suppression mode on BVI interface in the PE router.

  Example:

```
Router# configure
Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# vrf vrf1
Router(config-if)# ipv6 nd evpn-proxy mode partial-suppression
Router(config-if)# ipv6 enable
Router(config-if)# exit
Router(config)# commit
```

Step 2 Configure the L2VPN bridge domain and associate the BVI with the EVI.

Example:

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd1
Router(config-l2vpn-bg-bd)# interface HundredGigE0/0/0/0
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# routed interface BV1
Router(config-l2vpn-bg-bd)# evi 1
Router(config-l2vpn-bg-bd)# exit
Router(config-l2vpn-bg)# exit
Router(config-l2vpn)# exit
Router(config)# commit
```

Step 3 Verify the configuration with the **show running-config** command.

Example:

```
Router# sh running-config interface BVI1
interface BVI1
host-routing
vrf vrf1
ipv6 nd evpn-proxy mode full-suppression
ipv6 address 2001:DB8::1/32
ipv6 enable
mac-address 0011.1111.1111
!
```

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

MAC and MAC-IP Sequence Numbers

The IRB gateway device assigns, manages, and advertises sequence numbers that are associated with the locally learnt MAC routes through hardware learning, and the locally learnt MAC-IP routes through ARP.

Synchronized MAC and MAC-IP Sequence Numbers

In a host that is multi-homed to two ToRs, the locally learnt MAC and MAC-IP routes are synchronized across the two multi-homing peers through Route Type 2 learnt routes with a local ESI. So a device could have either MAC and MAC-IP, or both of them, learnt through both synchronized and local learning. Sequence numbers are synchronized across local and synchronized routes, because of which the sequence number that is advertised from the two ToRs for a given route is always the same.In certain situations, remote-sync route with same ESI can have a higher sequence number than a local route. In such a case, the local route sequence number is bumped up to match remote-sync route sequence number.

Local Sequence Number Updates

Host mobility is triggered when a local route is learnt while a remote route already exists. When mobility occurs, the local route is assigned a sequence number that is one higher than the existing remote route. This new local route is then advertised to the rest of the network.

Best Route Selection after Host Movement

When a host moves, the EVPN-PE at the new location of the host generates and advertises a higher sequence route to the network. When a higher sequence number route is received, as per RFC 7432, it is considered as the new best route and it is used for forwarding traffic. Best route selection is done for both MAC and MAC-IP routes.

Stale Route Deletion after a Host Movement

After a host moves from local to remote ESI, if a remote route from a different ESI is received and if a local route for the same host with a lower sequence number exists, then the local route is deleted and is withdrawn from the network.

The new higher sequence number remote MAC route is now considered best and is used to forward traffic. An ARP probe is sent to the host at the old local location. Because the host is at new remote location, probe will not succeed, resulting in clearing old local MAC-IP route.

Host Movement Detection through GARP

If a host sends a Gratuitous ARP (GARP) at its new location after a movement, the local MAC and local MAC-IP learning independently trigger mobility for both routes.

Host Move Detection with Silent Host

If a host does not send a GARP or a data packet at its new location following a move, the aging of the local MAC at the old location triggers mobility for both routes.

Host Move Detection without GARP with Data Packet

If the host does not send a GARP following a move, a data packet from the host triggers a proactive ARP probe to discover host MAC-IP and trigger mobility for this host across the overlay.

Duplicate MAC Detection

Duplicate MAC detection and freezing is supported as per RFC 7432.

Detection: Duplicate detection and recovery parameters are configurable. The default configuration is five times in 180 seconds and route freezing after three duplicate cycles. With the default configuration, when a host moves five times in 180 seconds, it is marked as duplicate for 30 seconds. Route advertisement for hosts in Duplicate state is suppressed. Host is taken out of duplicate state after 30 seconds. After a host is detected as duplicate for 3 times, on the fourth duplicate cycle, the host is permanently frozen. All route advertisements are suppressed for the frozen hosts.

In multi-homed hosts, a MAC is not necessarily learnt locally but is learnt through synchronization. Duplicate detection is supported for both local and remote-sync hosts. Remote-sync routes are differentiated from remote routes.

MAC-IP Handling: If the MAC route is in duplicate or frozen state, the corresponding local MAC-IP is updated, except that the route deletes are not withheld.

Duplicate State Handling: When a host is in duplicate state, route advertisements are suppressed. However, local routes are programmed in hardware so that traffic on local EVPN-PE is forwarded to the local host.

Recovery: It is possible to unfreeze permanently frozen hosts. The following is the recommended procedure to clear frozen hosts:

- Shutdown the host which is causing duplicate traffic.
- Use the **clear l2route evpn frozen-mac frozen-flag** command to clear the frozen hosts.

Configure EVPN IRB

Perform these steps to configure a bridge domain. You can perform this task over an MPLS core, and from Release 25.4.1, you can perform this task over 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.

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

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 A device can contain up to 128K MAC address entries. A bridge domain on a device can contain up to 65K MAC address entries.

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 Segment Routing.

Example:

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

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
```

The **label mode per-vrf** setting ensures the IPv6 VRF label is used as the second label in EVPN Route Type 2 (MAC+IP) and Route Type 5 advertisements.

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 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


/* Configure CEF */ [Required for dual homing]
Router# configure
Router(config)# cef adjacency route override rib
```

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
```

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.

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 */
Router# configure
Router(config)# evpn
Router(config-evpn)# host ipv4-address duplicate-detection
Router(config-evpn-host-ipv4-addr)# move-count 2
Router(config-evpn-host-ipv4-addr)# freeze-time 10
Router(config-evpn-host-ipv4-addr)# retry-count 2
Router(config-evpn-host-ipv4-addr)# commit

/* Ipv6 Address Duplicate Detection Configuration */
Router# configure
Router(config)# evpn
Router(config-evpn)# host ipv6-address duplicate-detection
Router(config-evpn-host-ipv6-addr)# move-count 2
Router(config-evpn-host-ipv6-addr)# freeze-time 10
Router(config-evpn-host-ipv6-addr)# retry-count 2
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.

```
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;
        (S)=Peer Sync; (Spl)=Split; (Rcv)=Recd;
        (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
```

Related Topics

- Duplicate IP Address Detection, on page 58

Associated Commands

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

EVPN Automatic Unfreezing of MAC and IP Addresses

The EVPN Automatic Unfreezing of MAC and IP Addresses feature unfreezes the permanently frozen MAC and IP addresses automatically. This feature provides a configurable option to enable a MAC or IP address to undergo infinite duplicate detection and recovery cycles without being frozen permanently. The MAC or IP address is permanently frozen when duplicate detection and recovery events occur three times within a 24-hour window. If any of the duplicate detection events happen outside the 24-hour window, the MAC or IP address undergoes only one duplicate detection event and all previous events are ignored.

Use the **infinity** keyword to prevent freezing of the duplicate MAC or IP address permanently.

Example

```
host ipv4-address duplicate-detection retry-count infinity
host ipv6-address duplicate-detection retry-count infinity
host mac-address duplicate-detection retry-count infinity
```

Use the **no** form of the above command to enable permanent freezing of MAC or IP address after the default retry count.

Example

```
no host ipv4-address duplicate-detection retry-count infinity
no host ipv6-address duplicate-detection retry-count infinity
no host mac-address duplicate-detection retry-count infinity
```

The 24-hour check for consecutive duplicate detection and recovery events before permanent freezing is enabled by default. Use the **reset-freeze-count-interval** keyword to configure a non-default interval after which the retry-count is reset. The range is from is 1 hour to 48 hours. The default is 24 hours.

Example

```
host ipv4-address duplicate-detection reset-freeze-count-interval 20
```

```
host ipv6-address duplicate-detection reset-freeze-count-interval 20
host mac-address duplicate-detection reset-freeze-count-interval 20
```

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 EVPN Automatic Unfreezing of MAC or IP Address

Infinite duplicate detection and recovery is disabled by default. However, you can enable it using the following configuration.

Configuration Example

```
/* IPv4 Address Duplicate Detection Configuration */
Router# configure
Router(config)# evpn
Router(config-evpn)# host ipv4-address duplicate-detection
Router(config-evpn-host-ipv4-addr)# move-count 5
Router(config-evpn-host-ipv4-addr)# move-interval 180
Router(config-evpn-host-ipv4-addr)# freeze-time 30
Router(config-evpn-host-ipv4-addr)# retry-count 3
Router(config-evpn-host-ipv4-addr)# reset-freeze-count-interval 24
Router(config-evpn-host-ipv4-addr)# commit

/* IPv6 Address Duplicate Detection Configuration */
Router# configure
Router(config)# evpn
Router(config-evpn)# host ipv4-address duplicate-detection
Router(config-evpn-host-ipv6-addr)# move-count 5
Router(config-evpn-host-ipv6-addr)# move-interval 180
Router(config-evpn-host-ipv6-addr)# freeze-time 30
Router(config-evpn-host-ipv6-addr)# retry-count 3
Router(config-evpn-host-ipv6-addr)# reset-freeze-count-interval 24
Router(config-evpn-host-ipv6-addr)# commit

/* MAC Address Duplicate Detection Configuration */
Router# configure
Router(config)# evpn
Router(config-evpn)# host MAC-address duplicate-detection
Router(config-evpn-host-mac-addr-dup-detection)# move-count 5
Router(config-evpn-host-mac-addr-dup-detection)# freeze-time 30
Router(config-evpn-host-mac-addr-dup-detection)# move-interval 180
Router(config-evpn-host-mac-addr-dup-detection)# retry-count infinite
Router(config-evpn-host-mac-addr-dup-detection)# reset-freeze-count-interval 24
Router(config-evpn-host-mac-addr-dup-detection)# commit
```

Running Configuration

This section shows the EVPN automatic unfreezing of MAC or IP address running configuration.

```
evpn
 host ipv4-address duplicate-detection
  move-count 5
```

```
  freeze-time 30
  retry-count 3
  reset-freeze-count-interval 24
 !
evpn
 host ipv6-address duplicate-detection
  move-count 5
  freeze-time 30
  retry-count 3
 !
evpn
 host mac-address duplicate-detection
  move-count 5
  freeze-time 30
  move-interval 180
  reset-freeze-count-interval 24
 !
```

Verification

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

```
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        Infinite   24
IPv4   False      30       5       180        3          24
IPv6   False      30       5       180        3          24
--------------------------------------------------------------------------
```

Related Topics

EVPN Automatic Unfreezing of MAC and IP Addresses, on page 60

Associated Commands

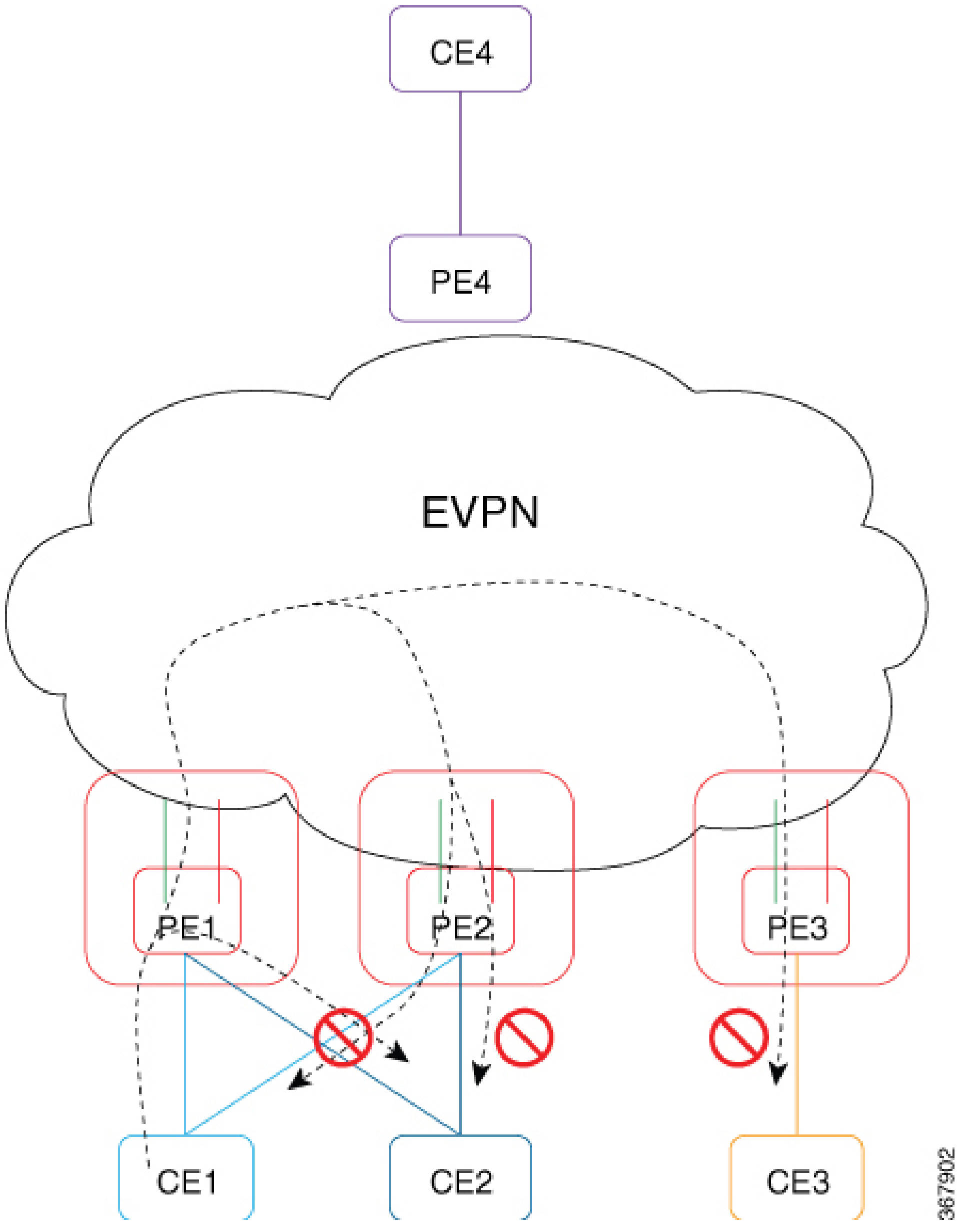
- host ipv4-address duplicate-detection
- host ipv6-address duplicate-detection
- host mac-address duplicate-detection
- show l2route summary

EVPN E-Tree

The EVPN E-Tree feature provides a rooted-multipoint Ethernet service over MPLS core. The EVPN Ethernet Tree (E-Tree) service enables you to define attachment circuits (ACs) as either a root site or a leaf site, which helps in load balancing and avoiding loops in a network.

In this topology, consider PE1, PE2, and PE3 as leaf ACs, and PE4 as root AC. Root ACs can communicate with all other ACs. Leaf ACs can communicate with root ACs but not with other leaf ACs with either L2 unicast or L2 BUM traffic. If a PE is not configured as E-Tree leaf, it is considered as root by default. This feature only supports leaf or root sites per PE.

Figure 7: EVPN E-Tree



E-Tree leaf is configured for each EVI Bridge Domain (BD). Root and leaf EVI of BD exports or imports single Routed Targets (RTs). The configuration of E-Tree leaf per EVI implies the following:

- All ACs inherit the leaf indicator.
- Split-horizon group between the ACs (leaf) on same EVI is enabled automatically.
- Each PE leaf advertises per Ethernet Segment per Ethernet Auto Discovery Route (ES-EAD), Ethernet Segment Identifier (ESI), ES-EAD ESI 0 route with leaf indicator and E-Tree label to BGP.

- All local MACs learned under this EVI are re-advertised to BGP with E-Tree leaf indicator.
- Each PE maintains a list of remote PEs.

Note
- If you modify the E-Tree leaf configuration, all the locally learned MAC addresses are flushed out. All the locally learned MAC addresses are flushed out even when bridge port's "encapsulation" or "rewrite" on sub-interface, or "split-horizon group" configuration is modified under the bridge port.
- A BVI interface configured in the bridge domain always exhibits the root behaviour.

Unicast Rules

The following table describes the unicast rules upon reception of type-2 MAC route on root and leaf.

MAC Route Received	MAC Route Handling
MAC address with non-local ESI from root EVI (BD)	Remote MAC address.
MAC address with local ESI from root EVI (BD)	MAC address synhronization, re-originate.
MAC address with non-local ESI from leaf EVI (BD)	Remote MAC address. Remote MAC route with leaf indicator is dropped.
MAC address with local ESI from leaf EVI (BD)	MAC address synhronization, re-originate. MAC address points to the local AC. Upon local AC failure, synchronization MAC route becomes a remote MAC route. Remote MAC route with leaf indicator is dropped as opposed to pointing to a peering PE.

Multicast Rules

Multicast is used to discover the leaf in the network when:

- RT-1 ES-EAD ESI-0 route with E-Tree extended community is sent per EVI (BD) to indicate to other network PEs which EVIs are setup as E-Tree leaf.
- RT-1 ES-EAD ESI-0 route with E-Tree extended community route and RT-3 IMCAST route are received on a leaf EVI (BD).

Note Per local EVI (BD) split-horizon group prevents local AC to AC traffic flow.

Communication between CE1 and CE4 (Inter-subnet)

1. CE1 sends an ARP request to its gateway, which is IRB interface. CE1 resolves the BVI IP address.
2. ARP request reaches the bridge domain on PE1. It learns the entry and floods it.

3. ARP requests to all remote PEs that have been pruned is dropped. It is replicated to all root remote PEs and to local BVI interface.
4. BVI interface on PE1 sends an ARP response to CE1 using its BVI IP address and BVI MAC address.
5. At the same time, since host routing is configured, PE1 advertises CE1 host route through EVPN using route type-2.
6. After receiving type-2 route, different rules apply based on the PE. After receiving route type-2 on:
   a. PE2: MAC and IP address of ESI match local ESI. Program MAC address as synchronization route. Program IP address in RIB to point to PE1, but MAC address points to CE1. Upon link failure to CE1, MAC address is marked as dropped in the hardware instead of pointing to peering PE1.
   b. PE3: MAC and IP address of ESI are not local. Since local EVI (BD) is leaf, MAC address is marked as dropped in the hardware. Program IP address in RIB pointing to PE1.
   c. PE4: MAC and IP address of ESI are not local. Since local EVI (BD) is root, program MAC as remote. Program IP address in RIB pointing to PE1.
7. PE4 is aware of CE1. CE1 and CE4 communicate with each other.
8. For example, a routing packet coming from CE4 reaches PE4. An IP lookup is performed. PE1 is found as the best destination due to the host route /32. The packet is forwarded to PE1.
9. On PE1, an IP lookup is performed. The BVI interface is found. The packet is encapsulated with CE1 as destination MAC address as learned by ARP. Source MAC address remains as the BVI MAC address. Destination MAC address lookup is performed in the corresponding bridge domain. The packet is forwarded to proper output interface.

Note If CE4 sends packet to CE1 before CE1 starts communication, the packet may go to peering PE2. GLEAN adjacency is affected and traffic is dropped until it is resolved. To resolve the entry, PE2 BVI interface starts probing.

1. ARP probing coming from BVI is sent to all ACs and EVI as well (L2 stretch).
2. PE1 and PE3 receive the ARP probe from EVI interface and replicate to all local ACs. CE1 sends ARP reply where PE1 BVI interface accepts it since IRB on all the leafs are configured in a distributed anycast gateway.

Communication between CE1 and CE3 (Intra-subnet)

1. CE1 and CE3 are within the same subnet.
2. CE1 sends an ARP request to CE3.
3. ARP request reaches the bridge domain on PE1. It learns the entry and floods it.
4. ARP requests for all remote PEs that have been pruned is dropped. It is replicated to all root remote PEs and to local BVI interface.
5. CE3 does not receive ARP request from CE1. CE1 with does not communicate with CE3.

6. If you want CE1 and CE3 to communicate within intra-subnet, then you must configure local_proxy_arp under BVI interface on both local and remote PEs.

Communicatiion between CE1 and CE2 (Intra-subnet)

1. CE1 and CE2 are within the same subnet.
2. CE1 sends an ARP request to CE2.
3. ARP request reaches the bridge domain on PE1. It learns the entry and floods it.
4. ARP requests for all remote PEs that have been pruned is dropped. It is not replicated to any local ACs due to common split-horizon group.
5. CE2 does not receive ARP request from CE1. CE1 does not communication with CE2.

Note Communication between local CE1 and remote CE1:

- The BUM traffic from local CE1 on PE1 to remote CE1 on PE2 is dropped as PE2 is pruned.
- The BUM traffic from local CE1 on PE1 to local CE1 on PE1 in the case of AC-Aware VLAN bundling feature is dropped due to ESI-filtering.

Configure EVPN E-Tree

Perform this task to configure EVPN E-Tree feature on the leaf PEs.

```
/* Configure EVPN E-Tree service on PE1 and PE2 */

Router# configure
Router(config)# evpn
Router(config-evpn)# evi 1
Router(config-evpn-evi)# etree leaf
```

Configuration Example

```
/* Configure EVPN Multihoming on PE1 and PE2*/

Router# configure
Router(config)# evpn
Router(config-evpn)# interface bundle-Ether 1121
Router(config-evpn-ac)# ethernet-segment  identifier type 0  20.00.00.00.00.00.00.11.21

/* Configure AC interface on PE1 and PE2*/

Router(config)# interface Bundle-Ether1121.1 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric

/* Configure BVI interface on PE1 and PE2 */

Router(config)# interface BVI1
```

```
Router(config-if)# host-routing
Router(config-if)# vrf vpn1
Router(config-if)# ipv4 address 192.0.2.1 255.255.255.0
Router(config-if)# proxy-arp
Router(config-if)# local-proxy-arp
Router(config-if)# ipv6 address 2001:DB8::1/32
Router(config-if)# mac-address 10.1111.aaaa
Router(config-if)# load-interval 30

/* Configure the bridge on PE1 and PE2 */

Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd1
Router(config-l2vpn-bg-bd)# interface Bundle-Ether1121.1
Router(config-l2vpn-bg-bd-ac)# exit
Router(config-l2vpn-bg-bd)# routed interface BVI1
Router(config-l2vpn-bg-bd-bvi)# exit
Router(config)# evpn
Router(config-evpn)# evi 1
Router(config-evpn-evi)# etree leaf
Router(config-evpn-instance)# commit
```

Running Configuration

This section shows EVPN E-Tree running configuration.

```
/* EVPN E-Tree running configuration on PE1 *
evpn
 evi 1
  etree
    leaf
  !
 !
 interface Bundle-Ether1121
  ethernet-segment
   identifier type 0 20.00.00.00.00.00.00.11.21


l2vpn
 bridge group bg1
  bridge-domain bd1
   interface Bundle-Ether1121.1
   routed interface BVI1
  !
  evi 1

interface Bundle-Ethe1121.1
l2transport
 encapsulation dot1q 1
 rewrite ingress tag pop 1 symmetric
 !
!
interface BVI1
 host-routing
 vrf vpn1
 ipv4 address 192.0.2.1 255.255.255.0
 proxy-arp
 local-proxy-arp
 ipv6 address 2001:DB8::1/32
 mac-address 10.1111.aaaa
 load-interval 30
```

```
 !
!

/* EVPN E-Tree running configuration On PE2 */

evpn
 evi 1
  etree
   leaf
  !
 !
 interface Bundle-Ether1121
  ethernet-segment
   identifier type 0 20.00.00.00.00.00.00.11.21

l2vpn
 bridge group bg1
  bridge-domain bd1
   interface Bundle-Ether1121.1
   routed interface BVI1
  !
  evi
 !
interface Bundle-Ethe1121.1
l2transport
 encapsulation dot1q 1
 rewrite ingress tag pop 1 symmetric
 !
!
interface BVI1
 host-routing
 vrf vpn1
 ipv4 address 192.0.2.1 255.255.255.0
 proxy-arp
 local-proxy-arp
 ipv6 address 2001:DB8::1/32
 mac-address 10.1111.aaaa
 load-interval 30
 !
!
```

Verification

The show output given in the following section display the details of the EVPN E-Tree configuration.

```
Router#show bgp l2vpn evpn rd 10.0.0.1:0
Route Distinguisher: 10.0.0.1:0
*> [1][10.0.0.1:1][0000.0000.0000.0000.0000][4294967295]/184
                      0.0.0.0                                0 i
*> [1][10.0.0.1:2][0000.0000.0000.0000.0000][4294967295]/184
                      0.0.0.0                                0 i

Each RT-1 ES0 has up to 200 RTs. Two RT-1 ES0 is displayed if you have 250 RTs.
```

The following output shows Leaf excom advertised in RT-1 ES0.

```
Router#show bgp l2vpn evpn rd 10.0.0.1:0
[1][10.0.0.1:1][0000.0000.0000.0000.0000][4294967295]/184
Extended community: EVPN E-TREE:0x00:824348 RT:100:1 RT:100:2 RT:100:3 RT:100:4 RT:100:5
RT:100:10 RT:100:11
```

```
RT:100:12 RT:100:13 RT:100:14 RT:100:15 RT:100:16 RT:100:17 RT:100:18 RT:100:19 RT:100:20
RT:100:21 RT:100:22 RT:100:23
RT:100:24 RT:100:25 RT:100:26 RT:100:27 RT:100:28 RT:100:29 RT:100:30 RT:100:31 RT:100:32
RT:100:33 RT:100:34 RT:100:35
RT:100:36 RT:100:37 RT:100:38 RT:100:39 RT:100:40 RT:100:41 RT:100:42 RT:100:43 RT:100:44
RT:100:45 RT:100:46 RT:100:47
 RT:100:48 RT:100:49 RT:100:50
```

The following output shows RT-2 of MAC advertisement.

```
Router#show bgp l2vpn evpn rd 10.0.0.1:1 [2][1][48][0011.1100.0001][0]/104
Paths: (2 available, best #1)
  Advertised to peers (in unique update groups):
    172.16.0.1
  Path #1: Received by speaker 0
  Advertised to peers (in unique update groups):
    172.16.0.1
  Local
    0.0.0.0 from 0.0.0.0 (10.0.0.1)
      Origin IGP, localpref 100, valid, redistributed, best, group-best, import-candidate,
 rib-install
      Received Path ID 0, Local Path ID 1, version 315227
      Extended community: SoO:192.168.0.1:1 EVPN E-TREE:0x01:0 RT:100:1
      EVPN ESI: 0020.0000.0000.0000.1121
```

The following output shows one RT-2 of MAC address and IP address advertisement.

```
Router#show bgp l2vpn evpn rd 10.0.0.1:1 [2][1][48][0011.1100.0001][32][101.0.1.103]/136
Tue Oct  2 16:44:26.755 EDT
BGP routing table entry for [2][1][48][0011.1100.0001][32][101.0.1.103]/136, Route
Distinguisher: 10.0.0.1:1
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker             313139      313139
    Local Label: 820002
Last Modified: Oct  2 13:26:08.477 for 03:18:18
Paths: (2 available, best #1)
  Advertised to peers (in unique update groups):
    172.16.0.1
  Path #1: Received by speaker 0
  Advertised to peers (in unique update groups):
    172.16.0.1
  Local
    0.0.0.0 from 0.0.0.0 (10.0.0.1)
      Second Label 825164
      Origin IGP, localpref 100, valid, redistributed, best, group-best, import-candidate,
 rib-install
      Received Path ID 0, Local Path ID 1, version 313139
     Extended community: Flags 0xe: SoO:192.168.0.1:1 EVPN E-TREE:0x01:0 RT:100:1 RT:991:1

      EVPN ESI: 0020.0000.0000.0000.1121
```

The following output shows aggregation of RT-3 inclusive-multicast and RT-1 ES0 routes in EVPN.

```
Router#show evpn evi vpn-id 1 inclusive-multicast detail
1          MPLS   0          192.168.0.1
    TEPid  : 0x02000001
    PMSI Type: 0
    Nexthop: 192.168.0.1
    Label  : 810120
    Source : Remote
    E-Tree: Leaf
1          MPLS   0          10.0.0.1
```

```
    TEPid  : 0xffffffff
    PMSI Type: 6
    Nexthop: ::
    Label  : 820120
    Source : Local
    E-Tree: Leaf
1          MPLS   0          172.16.0.1
    TEPid  : 0x02000003
    PMSI Type: 0
    Nexthop: 172.16.0.1
    Label  : 840120
    Source : Remote
    E-Tree: Root
```

Related Topics

- EVPN E-Tree, on page 62

Associated Commands

- etree leaf
- show bgp l2vpn evpn rd

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::

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.

As you are using EVPN E-Tree with RT constraints and rt-leaf indicator set, the rt-leaf configuration causes EVPN to add the ES-import-RT to the mac-only RT-2s to support All-Active syncing for a Bundle Multihomed by Leafs. This is required when using RT constraints, otherwise the PE can end up flooding unicast traffic to its local ACs forever.

```
evpn evi 2001
  etree
   rt-leaf
```

```
RP/0/RP0/CPU0:router# show evpn evi vpn-id 2001 mac
VPN-ID  Encap  MAC address    IP address    Nexthop              Label    SID
------  -----  -------------- ----------    -------------------  -----    ---
2001    MPLS   0000.0100.0003 ::            Bundle-Ether11.2002   26064
```

```
RP/0/RP0/CPU0:router# show l2route evpn mac all detail

Flags: (Stt)=Static; (L)=Local; (Lp)=Local-Proxy;
       (R)=Remote;N)=No Redistribution; (Rtr)=Router MAC;
       (B)=Best Route;(S)=Peer Sync; (Spl)=Split; (Rcv)=Recd;
       (D)=Duplicate MAC; (Z)=Frozen MAC;(Sfa)=Single Flow Active
       (A)=Access; (Gw)=Gateway;

Topo ID  Mac Address     Producer   Next Hop(s)               Seq No Flags   Slot       ESI
  Opaque Data Type  Opaque Data Len  Opaque Data Value
-------  --------------  --------   ------------------------  ------ -----   --------   ---
  ----------------  ---------------  -----------------
16       0000.0100.0003  LOCAL      Bundle-Ether11.2002, N/A  0      BLRcv   0/0/CPU0   (F)
   N/A               N/A              N/A

Last Update: Tue Mar 28 12:58:00.730
```

```
Router# show evpn ethernet-segment interface bundle-Ether 11 carving private
Legend:
Ethernet Segment Id Interface Nexthops
0010.1010.1010.1010.1011 BE11 172.16.45.3
172.16.45.4
ES to BGP Gates : Ready
ES to L2FIB Gates : Ready
Main port :
Interface name : Bundle-Ether11
Interface MAC : bc2c.e654.b8dc
IfHandle : 0x20008034
State : Up
Redundancy : Not Defined
ESI ID : 0x2
ESI type : 0
Value : 0010.1010.1010.1010.1011
ES Import RT : 1010.1010.1010 (from ESI)
Source MAC : 0000.0000.0000 (N/A)
Topology :
Operational : MH, All-active
Configured : All-active (AApF) (default)
Service Carving : Auto-selection
Multicast : Disabled
Convergence : Reroute
Peering Details : 2 Nexthops
172.16.45.3 [MOD:P:00:T][2]
172.16.45.4 [MOD:P:7fff:T][0]
```

```
Router# show bgp l2vpn evpn rd [2][0][48][0000.0100.0003][0]/104 DET
BGP routing table entry for [2][0][48][0000.0100.0003][0]/104, Route Distinguisher:
172.16.45.4:2001
Versions:
Process bRIB/RIB SendTblVer
Speaker 5373 5373
Local Label: 26064 (no rewrite);
Flags: 0x00040001+0x00000000;
Last Modified: Jun 2 03:42:14.557 for 2d12h
Paths: (1 available, best #1)
Advertised to update-groups (with more than one peer):
0.2
Path #1: Received by speaker 0
Flags: 0x202002000504000b+0x00, import: 0x000, EVPN: 0x1
Advertised to update-groups (with more than one peer):
0.2
Local
0.0.0.0 from 0.0.0.0 (172.16.45.4), if-handle 0x00000000
Origin IGP, localpref 100, valid, redistributed, best, group-best,
import-candidate, rib-install
```

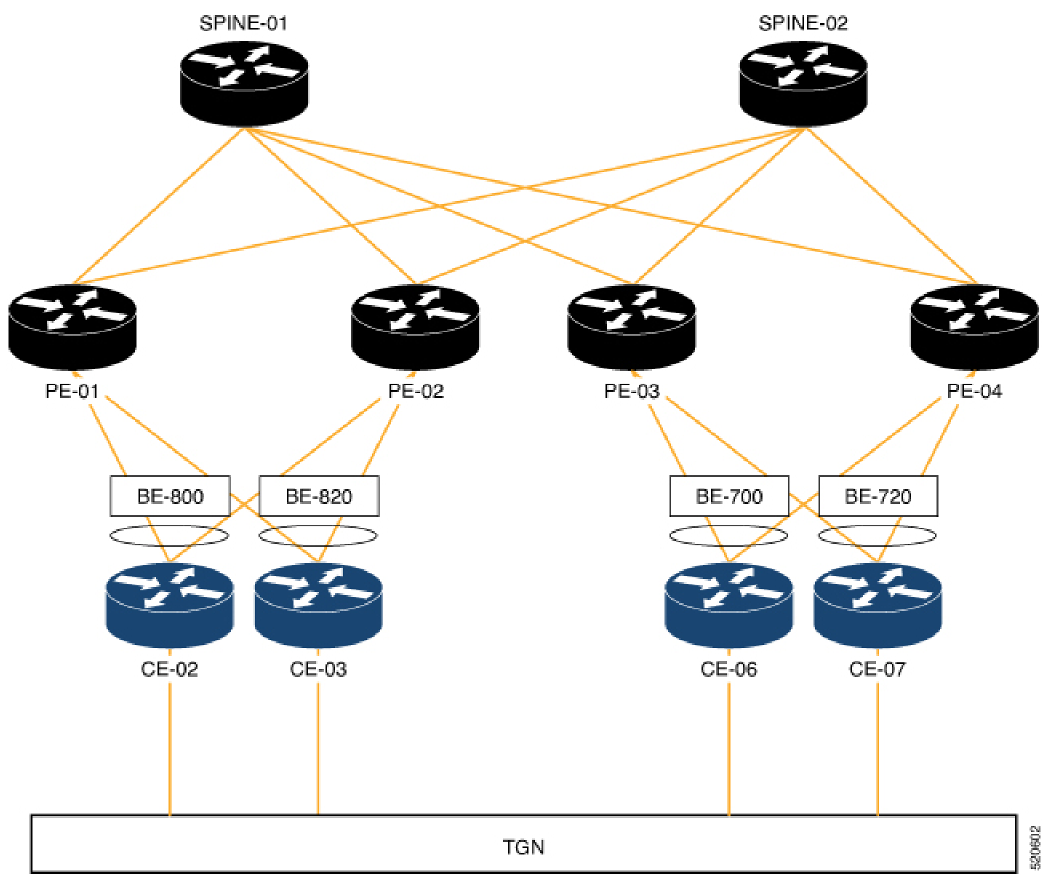
```
Received Path ID 0, Local Path ID 1, version 5367
Extended community: SoO:172.16.45.4:2001 EVPN ES Import:1010.1010.1010 EVI RT:fd84.0000.07d1
0x060e:0000.007d.2fff RT:2001:64900
EVPN ESI: 0010.1010.1010.1010.101

RP/0/RP0/CPU0:router# show evpn evi mac 0000.0100.0003 detail

Tue Apr 4 16:50:33.090 NZST

VPN-ID  Encap  MAC address     IP address  Nexthop              Label  SID
------  -----  --------------  ----------  -------------------  -----  ---
2001    MPLS   0000.0100.0003  ::          Bundle-Ether11.2002  26064

   Ethernet Tag                              : 0
   Multi-paths Resolved                      : False
   Multi-paths Internal label                : 0
   Local Static                              : No
   Remote Static                             : No
   Local Ethernet Segment                    : 0010.1010.1010.1010.1011
   Remote Ethernet Segment                   : N/A
   Local Sequence Number                     : 0
   Remote Sequence Number                    : N/A
   Local Encapsulation                       : MPLS
   Remote Encapsulation                      : N/A
   Local E-Tree                              : Leaf
   Remote E-Tree                             : Root
   Remote matching E-Tree RT                 : No
   Local AC-ID                               : 0x7d2fff
   Remote AC-ID                              : 0x0
```



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

- #unique_371

Associated Commands

- etree rt-leaf
- show l2route evpn mac all

EVPN E-Tree Per-PE (Scenario 1b)

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
EVPN E-Tree Per-PE (Scenario 1b)	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 an EVPN Instance (EVI) or for a given bridge-domain. By preventing communication among leaf ACs connected to the same PE and belonging to the same EVI, you can segregate traffic received and sent from different geographical locations. This segregation helps in load balancing traffic and avoiding traffic from going into loops in a network.

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 or EVI, 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 EVI or 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 1b

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

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.

For E-Tree with IRB, BVI interfaces are considered as a root site. However, if you configure the PE as a leaf site that has ACs with BVI, ingress filtering is performed instead of egress filtering as defined by Option B in RFC 8317. Tagging of ingress BUM traffic with **etree leaf** label is performed for the packets destined for a remote node.

Scenario 1b Behavior

- E-Tree leaf is configured per bridge domain or EVI. No leaf configuration means the bridge domain or EVI is a root.
- All ACs inherit E-Tree leaf designation from the bridge domain or EVI.
- Split-horizon group between ACs of a leaf is enabled automatically.
- All local MACs learned under the BD or EVI 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.
- EVPN E-Tree with IRB is also supported, but BVI interfaces are always treated as a root sites, even if BD or EVI itself is a leaf.

Restrictions

- If a BVI interface is part of a bridge domain, we recommend you to configure **etree leaf** under EVPN EVI configuration. When BVI is associated with an AC, etree leaf under the bridge domain is not supported due to hardware limitation.
- If an AC is not associated with BVI under a bridge domain, you can configure **etree leaf** under EVPN EVI or bridge domain configuration.
- Scenario 1a and Scenario 1b with IRB may not interopeate with any other flavors on E-Tree; this misconfiguration cannot be detected automatically.
- For non-supported interfaces, such as VNI, BVI, and PW, are not be marked as a leaf when the BD is configured as a leaf.
- You cannot configure a BD as E-Tree leaf when AC is a BVI interface. If an IRB interface is required, you must use the Cisco implementation (which is non-RFC compliant) of E-Tree where the EVI is configured as an E-Tree leaf.

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 bg_201
Router(config-l2vpn-bg)# bridge-domain bd_201
Router(config-l2vpn-bg-bd)# etree leaf
Router(config-l2vpn-bg-bd)# interface Bundle-Ether3501.3601
Router(config-l2vpn-bg-bd-ac)# evi 201
Router(config-l2vpnbg-bd-evi)# commit
```

Configure EVPN E-Tree leaf per EVI.

```
Router# configure
Router(config)# evpn
Router(config-evpn)# evi 200
Router(config-evpn-instance)# bgp route-target import 64600:200
Router(config-evpn-instance)# bgp route-target export 64600:200
Router(config-evpn-instance)# etree leaf
Router(config-evpn-instance)# control-word-disable
Router(config-evpn-instance)# advertise-mac
Router(config-evpn-instance-mac)# commit
```

Running Configuration

Note The E-Tree configuration under EVI is required only when BVI is used in the BD.

```
/* EVPN E-Tree leaf per bridge domain */
l2vpn
 bridge group bg_201
  bridge-domain bd_201
   etree
```

```
    leaf
   !
   interface Bundle-Ether3501.3601
   !
   evi 201
   !
  !
/* EVPN E-Tree leaf per EVI */
evpn
 evi 200
  bgp
   route-target import 64600:200
   route-target export 64600:200
  !
  etree
   leaf
  !
  control-word-disable
  advertise-mac
  !
 !
/* EVPN E-Tree with BVI */
l2vpn
 bridge group bg1
  bridge-domain bd1
   interface Bundle-Ether200.1
   !
   interface Bundle-Ether201.1001
   !
   routed interface BVI1
    split-horizon group core
   !
   evi 200
   !
  !
 !
```

Verification

Verify that the etree leaf is configured per bridge domain.

```
Router# show l2vpn bridge-domain bd-name bd_201 detail
Legend: pp = Partially Programmed.
Bridge group: etree_bg, bridge-domain: bd_201, id: 162, state: up, ShgId: 0, MSTi: 0
  Coupled state: disabled
  VINE state: EVPN Native
  MAC learning: enabled
  MAC withdraw: enabled
    MAC withdraw for Access PW: enabled
    MAC withdraw sent on: bridge port up
    MAC withdraw relaying (access to access): disabled
  Flooding:
    Broadcast & Multicast: enabled
    Unknown unicast: enabled
  MAC aging time: 300 s, Type: inactivity
  MAC limit: 64000, Action: none, Notification: syslog
  MAC limit reached: no, threshold: 75%
  MAC port down flush: enabled
  MAC Secure: disabled, Logging: disabled
  Split Horizon Group: none
  E-Tree: Leaf
  Dynamic ARP Inspection: disabled, Logging: disabled
  IP Source Guard: disabled, Logging: disabled
```

```
  DHCPv4 Snooping: disabled
  DHCPv4 Snooping profile: none
  IGMP Snooping: disabled
  IGMP Snooping profile: none
  MLD Snooping profile: none
  Storm Control: disabled
  Bridge MTU: 1500
  MIB cvplsConfigIndex: 163
  Filter MAC addresses:
  P2MP PW: disabled
  Multicast Source: Not Set
  Create time: 25/10/2021 15:50:01 (00:50:39 ago)
  No status change since creation
  ACs: 1 (0 up), VFIs: 0, PWs: 0 (0 up), PBBs: 0 (0 up), VNIs: 0 (0 up)
  List of EVPNs:
    EVPN, state: up
      evi: 1 (MPLS)
      XC ID 0x8000009f
     Statistics:
       packets: received 0 (unicast 0), sent 0
       bytes: received 0 (unicast 0), sent 0
       MAC move: 0
  List of ACs:
    AC: Bundle-Ether3501.3601, state is up (Segment-up)
      Type VLAN; Num Ranges: 1
      Rewrite Tags: []
      VLAN ranges: [3601, 3601]
      MTU 1500; XC ID 0xa00001e0; interworking none; MSTi 2
      MAC learning: enabled
      Flooding:
        Broadcast & Multicast: enabled
        Unknown unicast: enabled
      MAC aging time: 300 s, Type: inactivity
      MAC limit: 64000, Action: none, Notification: syslog
      MAC limit reached: no, threshold: 75%
      MAC port down flush: enabled
      MAC Secure: disabled, Logging: disabled
      Split Horizon Group: enabled (inherited)
      E-Tree: Leaf (inherited)
      Dynamic ARP Inspection: disabled, Logging: disabled
      IP Source Guard: disabled, Logging: disabled
      DHCPv4 Snooping: disabled
      DHCPv4 Snooping profile: none
      IGMP Snooping: disabled
      IGMP Snooping profile: none
      MLD Snooping profile: none
      Storm Control: bridge-domain policer
      Static MAC addresses:
      PD System Data: AF-LIF-IPv4: 0x00000000  AF-LIF-IPv6: 0x00000000 FRR-LIF: 0x00000000

  List of Access PWs:
  List of VFIs:
  List of Access VFIs:


Router# show l2vpn bridge-domain bd-name bd1 detail
Legend: pp = Partially Programmed.
Bridge group: bg1, bridge-domain: bd1, id: 6, state: up, ShgId: 0, MSTi: 0
  Coupled state: disabled
  VINE state: EVPN-IRB
  MAC learning: enabled
  MAC withdraw: enabled
    MAC withdraw for Access PW: enabled
    MAC withdraw sent on: bridge port up
    MAC withdraw relaying (access to access): disabled
```

```
Flooding:
  Broadcast & Multicast: enabled
  Unknown unicast: enabled
MAC aging time: 300 s, Type: inactivity
MAC limit: 64000, Action: none, Notification: syslog
MAC limit reached: no, threshold: 75%
MAC port down flush: enabled
MAC Secure: disabled, Logging: disabled
Split Horizon Group: none
E-Tree: Leaf (inherited)
Dynamic ARP Inspection: disabled, Logging: disabled
IP Source Guard: disabled, Logging: disabled
DHCPv4 Snooping: disabled
DHCPv4 Snooping profile: none
IGMP Snooping: disabled
IGMP Snooping profile: none
MLD Snooping profile: none
Storm Control: disabled
Bridge MTU: 1500
MIB cvplsConfigIndex: 7
Filter MAC addresses:
P2MP PW: disabled
Multicast Source: Not Set
Create time: 25/10/2021 15:50:01 (00:47:35 ago)
No status change since creation
ACs: 3 (0 up), VFIs: 0, PWs: 0 (0 up), PBBs: 0 (0 up), VNIs: 0 (0 up)
List of EVPNs:
  EVPN, state: up
    evi: 200 (MPLS)
    XC ID 0x80000003
   Statistics:
     packets: received 0 (unicast 0), sent 0
     bytes: received 0 (unicast 0), sent 0
     MAC move: 0
List of ACs:
  AC: BVI1, state is up (Segment-up)
    Type Routed-Interface
    MTU 1514; XC ID 0x800007d3; interworking none
    BVI MAC address:
      0011.1111.1111
    Split Horizon Group: Core
    PD System Data: AF-LIF-IPv4: 0x00000000  AF-LIF-IPv6: 0x00000000 FRR-LIF: 0x00000000

  AC: Bundle-Ether200.1, state is up (Segment-up)
    Type VLAN; Num Ranges: 1
    Rewrite Tags: []
    VLAN ranges: [1, 1]
    MTU 1500; XC ID 0xa00000be; interworking none; MSTi 9
    MAC learning: enabled
    Flooding:
      Broadcast & Multicast: enabled
      Unknown unicast: enabled
    MAC aging time: 300 s, Type: inactivity
    MAC limit: 64000, Action: none, Notification: syslog
    MAC limit reached: no, threshold: 75%
    MAC port down flush: enabled
    MAC Secure: disabled, Logging: disabled
    Split Horizon Group: enabled (inherited)
    E-Tree: Leaf (inherited)
    Dynamic ARP Inspection: disabled, Logging: disabled
    IP Source Guard: disabled, Logging: disabled
    DHCPv4 Snooping: disabled
    DHCPv4 Snooping profile: none
    IGMP Snooping: disabled
```

```
      IGMP Snooping profile: none
      MLD Snooping profile: none
      Storm Control: bridge-domain policer
      Static MAC addresses:
      PD System Data: AF-LIF-IPv4: 0x00012678  AF-LIF-IPv6: 0x00012679 FRR-LIF: 0x00000000

    AC: Bundle-Ether201.1001, state is up (Segment-up)
      Type VLAN; Num Ranges: 1
      Rewrite Tags: []
      VLAN ranges: [1001, 1001]
      MTU 1500; XC ID 0xa000017c; interworking none; MSTi 9
      MAC learning: enabled
      Flooding:
        Broadcast & Multicast: enabled
        Unknown unicast: enabled
      MAC aging time: 300 s, Type: inactivity
      MAC limit: 64000, Action: none, Notification: syslog
      MAC limit reached: no, threshold: 75%
      MAC port down flush: enabled
      MAC Secure: disabled, Logging: disabled
      Split Horizon Group: enabled (inherited)
      E-Tree: Leaf (inherited)
      Dynamic ARP Inspection: disabled, Logging: disabled
      IP Source Guard: disabled, Logging: disabled
      DHCPv4 Snooping: disabled
      DHCPv4 Snooping profile: none
      IGMP Snooping: disabled
      IGMP Snooping profile: none
      MLD Snooping profile: none
      Storm Control: bridge-domain policer
      Static MAC addresses:
      PD System Data: AF-LIF-IPv4: 0x0001272c  AF-LIF-IPv6: 0x0001272d FRR-LIF: 0x00000000

  List of Access PWs:
  List of VFIs:
  List of Access VFIs:
```

DHCPv4 Relay on IRB

DHCPv4 Relay on Integrated Routing and Bridging (IRB) feature provides DHCP support for the end users in EVPN all-active multihoming scenario. This feature enables reduction of traffic flooding, increase in load sharing, optimize traffic, faster convergence during link and device failures, and simplification of data center automation.

DHCPv4 relay agent sends request packets coming over access interface towards external DHCPv4 server to request address (/32) allocation for the end user. DHCPv4 relay agent acts as stateless for end users by not maintaining any DHCPv4 binding and respective route entry for the allocated address.

DHCPv4 relay profiles are configured on bridge-group virtual interface (BVI) interfaces which act as access interfaces by integrating routing and bridge domains for the end users. It relays DHCPv4 requests from Layer 2 attachment circuit (AC) to external DHCP servers for host IPv4 addresses (/32).

Multihoming All-Active EVPN Gateways

Multihoming all-active EVPN gateways are configured with anycast IP address and MAC addresses. The Cisco routers have centralized L2 or L3 gateway. Based on native EVPN and MAC learning, IRB uses distributed anycast IP address and anycast MAC address. Static clients are configured with anycast gateway address as the default gateway. DHCP client sends DHCP requests for IP address allocation over the BVI interface. L2 access can be either single homing or multihoming, not all access protocols are supported with IRB. BVI IP address acts as a default gateway for the end user. The external DHCPv4 server provides this BVI interface IP address as default gateway in route options. No EVPN is configured on the Internet gateway.

EVPN IRB Route Distribution

In EVPN IRB DHCPv4, 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** and **relay information option vpn** 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 the IP addresses.

The IP address allocation for the end user at DHCPv4 server is based on **relay agent information** option (Remote-ID+ Circuit-ID) values. 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 from the same subnet of BVI interface.

After the DHCPv4 application receive the access side DHCPv4 packets over BVI interface based on **relay-option policy {encapsulate | drop | keep}** command, DHCPv4 application includes option-82 Relay-Agent Information, Remote-ID, and Circuit-ID for DHCPv4 Server.

The following table provides the attributes that qualify the DHCPv4 relay packets for the configured Relay-Information details. The information given in the table is used for configuring **relay-option policy {encapsulate | drop | keep}** command.

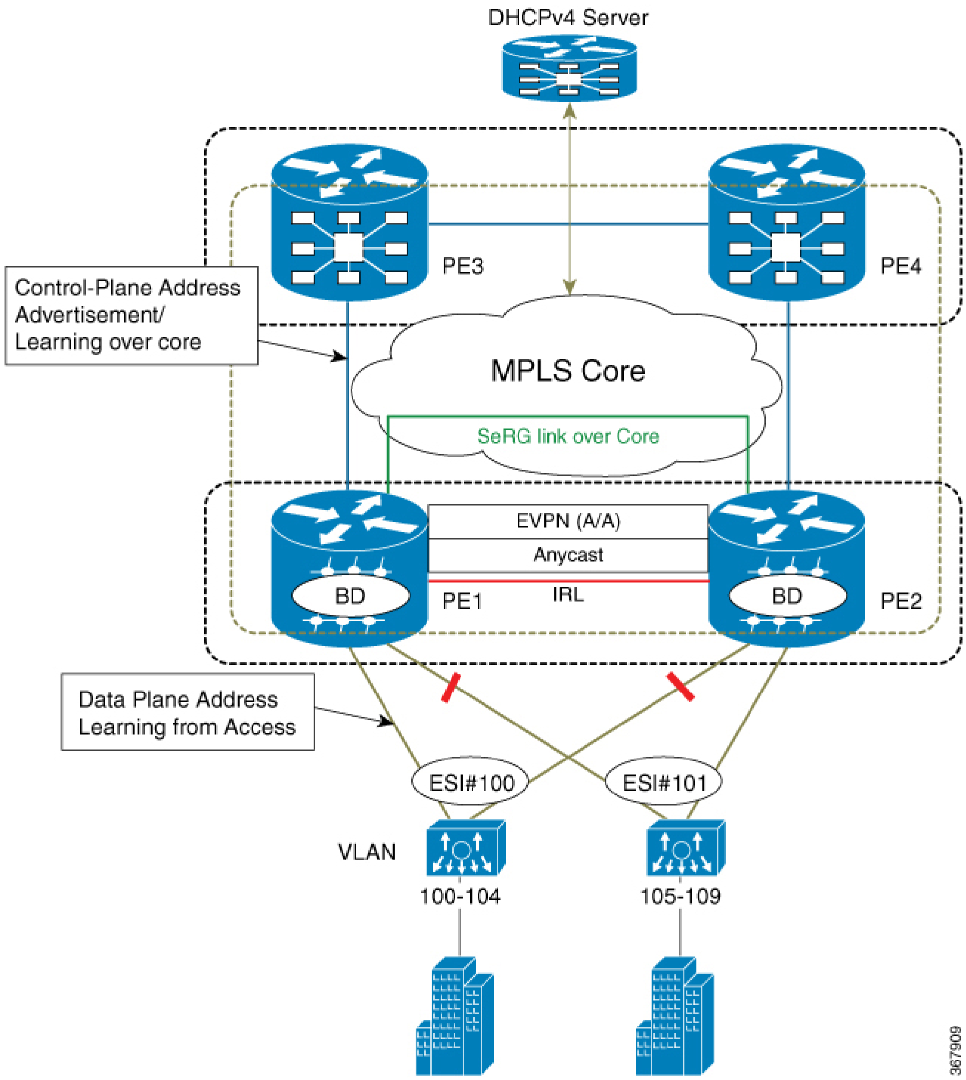
Relay-Option Policy	DHCPv4 Access Side Packet	Local Configuration	DHCPv4 Relay Packet Decision
Encapsulate	No Relay-Information	DHCPv4-Profile with Remote-ID L2Transport AC with Circuit-ID	Relay-Agent with Remote-ID and Circuit-ID
Encapsulate	Relay-Information (Remote-ID and Circuit-ID)	DHCPv4-Profile with Remote-ID L2Trasnsport AC with Circuit-ID	Override Relay-Agent Information with Local Configuration (Remote-ID and Circuit-ID)
Encapsulate	No Relay-Information	DHCPv4-Profile with Remote-ID and VPN-Information L2Transport AC with Circuit-ID	Relay-Agent with Remote-ID, Circuit-ID and VPN-Information
Keep	Relay-Information (Remote-ID and Circuit-ID)	No configuration	DHCPv4 Relay-Agent does not change any Relay-Information

Relay-Option Policy	DHCPv4 Access Side Packet	Local Configuration	DHCPv4 Relay Packet Decision
Keep	Relay-Information (Remote-ID and Circuit-ID)	DHCPv4-Profile with Remote-ID L2 Transport AC with Circuit-ID	DHCPv4 Relay-Agent does not change any Relay-Information
Keep	Relay-Information (Remote-ID and Circuit-ID)	DHCPv4-Profile with Remote-ID and VPN-Information L2 Transport AC with Circuit-ID	DHCPv4 Relay-Agent does not change any Relay-Information
Drop	Relay-Information (Remote-ID and Circuit-ID)	No configuration	Exclude Relay-Agent Information and include `None` in Relayed-Packet
Drop	Relay-Information (Remote-ID and Circuit-ID)	DHCPv4-Profile with Remote-ID L2 Transport AC with Circuit-ID	Exclude Relay-Agent Information and include `None` in Relayed-Packet
Drop	Relay-Information (Remote-ID and Circuit-ID)	DHCPv4-Profile with Remote-ID and VPN-Information L2 Transport AC with Circuit-ID	Exclude Relay-Agent Information and include `None` in Relayed-Packet

DHCP Request Forwarding Path

Clients broadcast requests to the access switch with DH-AA to EVPN PE routers. The access switch does load balancing. The load balancing configurations in access switch impacts 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 all-active PE routers are configured with the same IP address, BVI IP addresses cannot be used as DHCP relay source IP address. For DHCPv4 relay, access (BVI) interface is tied-up with relay profile. The device intercept packets are received over BVI interface and each relay profile is defined with Gateway IP Address (GIADDR), which acts as source IP address for initiated relayed packets towards DHCPv4 server. This GIADDR is unique across Top of Racks (ToRs) for respective BVI interfaces. Loopback interface with unique IPv4 address can be configured in VRF that is reachable to DHCP servers. Configuring DHCP relay source address is not supported.

Figure 8: PON behavior in handling DHCPv4 Server for EVPN All-Active Multihoming



PON behavior in handling DHCPv4 Server for EVPN All-Active Multihoming

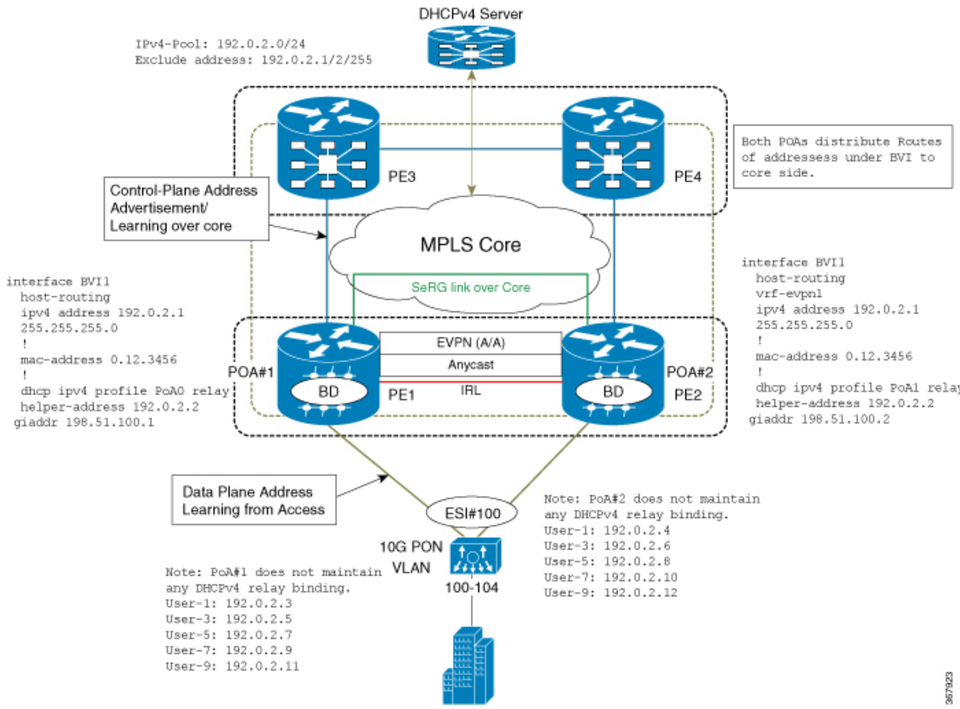
In this topology, PE1 and PE2 are edge routers for access side, which serve CEs (10G-OLT) over BVI interfaces by associating routing and bridging domains to process DHCPv4 packets. CEs (L2 OLT, PONs, any L2 domain switches) hashes the incoming control packets (DHCPv4 packets) towards port channels that are connected to respective PEs. The CEs leverage the hashing mechanism based on five tuples (src mac, dst mac, src-ip, dst-ip, L4 (tcp/udp) dst/src port) of packets that are received from the end user. Defines the forwarding mechanism by selecting the port channel on load balancing the control packets to respective PEs in dual-home active-active model.

DHPCv4 Relay Handling for EVPN and DHCPv4 Server in Default VRF

DHCPv4 relay over EVPN IRB and DHCPv4 servers resides in the same default VRFs. The DHCPv4 relay profiles are associated with helper-addresses of DHCPv4 address under default VRFs. In this particular scenario, PEs do not include any relay-agent information in relayed DHCPv4 packets towards DHCPv4 server.

However, DHCPv4 relay profile is defined in unique GIADDR across ToRs other than the anycast IRB address. Else, it is difficult for DHCPv4 server to perform address allocation for end user of not having link selection or subnet selection. The PEs include relay-agent information by including VPN information with VPN value as 0xFF.

Figure 9: DHPCv4 Relay Handling for EVPN and DHCPv4 Server in Default VRF

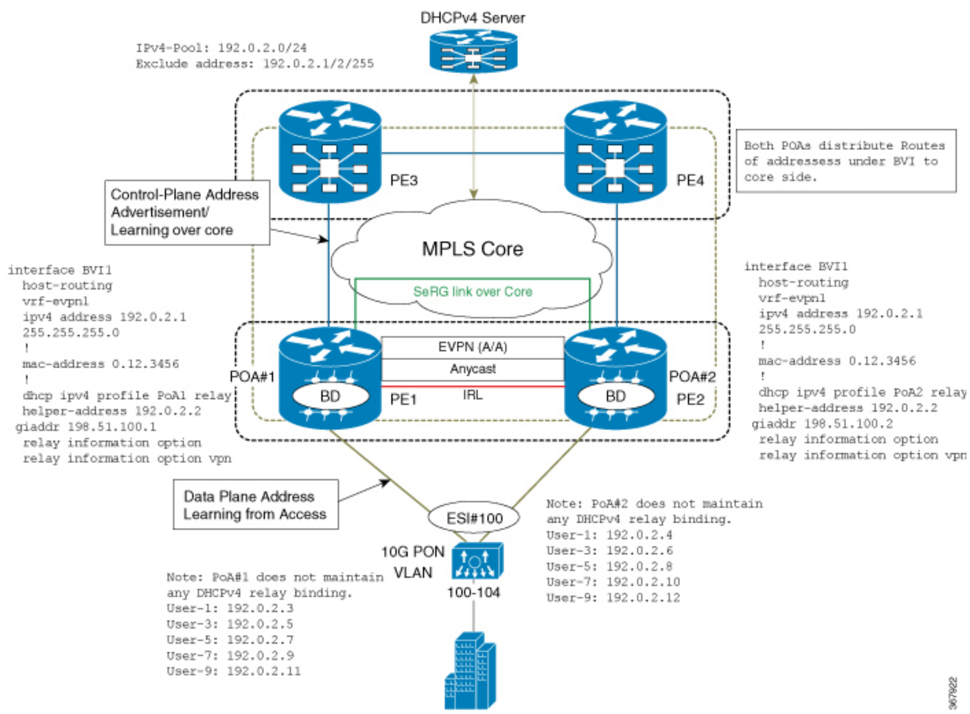


DHPCv4 Relay Handling for EVPN and DHCPv4 Server in Different VRF

DHCPv4 relay over EVPN IRB and DHCPv4 servers reside in different VRFs or DHCPv4 server has an unique GIADDR across ToRs which is different from the anycast IRB address. Else, it is difficult for DHCPv4 server to perform address allocation for end user of not having link selection or subnet selection. To ensure DHCPv4 server to provide address allocation from pool of subnet of related anycast IRB address of evpn, there is a way that ToRs of DHCPv4 relay agent intimate Virtual-Subnet-Selection (link-selection, server-id, vrf-id) by including Relay-Agent-Information (Option-82) in DHCPv4 relayed Discover and Request packets towards DHCPv4 Server.

In this topology, the 10G PON distributes equally the DHCP broadcast towards respective point of attachment (PoA) #1, #2, and packets are relayed to external DHCPv4 server.

Figure 10: DHPCv4 Relay Handling for EVPN and DHCPv4 Server in Different VRF



Configure DHCPv4 Relay on IRB

Perfrom these tasks to configure DHCPv4 Relay on IRB.

Configuration Example

```
/* PE1 configuration */

Router# configure
Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# vrf-evpn1
Router(config-if)# ipv4 address 192.0.2.1 255.255.255.0
Router(config)# mac-address 0.12.3456
!
Router# configure
Router(config)# dhcp ipv4
Router(config-dhcpv4)# profile PoA1 relay
Router(config-dhcpv4-relay-profile)# helper-address 192.0.2.2 giaddr 198.51.100.1
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)# commit

/* PE2 configuration */

Router# configure
Router(config)# interface BVI1
Router(config-if)# host-routing
```

```
Router(config-if)# vrf-evpn1
Router(config-if)# ipv4 address 192.0.2.1 255.255.255.0

Router(config)# mac-address 0.12.3456
!
Router# configure
Router(config)# dhcp ipv4
Router(config-dhcpv4)# profile PoA2 relay
Router(config-dhcpv4-relay-profile)# helper-address 192.0.2.2 giaddr 198.51.100.2
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)# commit
```

The following example shows a configuration of DHCPv4 relay agent to include Relay-Agent Information with Remote-ID and Circuit-ID. The Remote-ID is configured under DHCPv4-Relay-Profile, which is associated under BVI interface. DHCPv4 is configured with L2Transport ACs with Circuit-ID.

```
Dhcp ipv4
 Profile RELAY relay
  Relay information option remote-id format-type asci cisco
  Relay information policy encapsulate
 !

interface BE1.100 relay information option circuit-id format-type hex cisco
!
 interface bvi relay RELAY
!
```

Running Configuration

This section shows DHCPv4 relay on IRB running configuration.

```
/* PE1 Configuration */
interface BVI1
 host-routing
 vrf-evpn1
 ipv4 address 192.0.2.1 255.255.255.0
 !
 mac-address 0.12.3456
!
 dhcp ipv4 profile PoA1 relay
 helper-address 192.0.2.2 giaddr 198.51.100.1
 relay information option
 relay information option vpn-mode rfc

/* PE2 Configuration */
interface BVI1
 host-routing
 vrf-evpn1
 ipv4 address 192.0.2.1 255.255.255.0
 !
 mac-address 0.12.3456
!
 dhcp ipv4 profile PoA2 relay
 helper-address 192.0.2.2 giaddr 198.51.100.2
 relay information option
 relay information option vpn-mode rfc
```

Verification

Verify DHCPv4 Relay on IRB configuration.

```
/* Verify DHCPv4 relay statistics
Router# show dhcp vrf default ipv4 relay statistics

DHCP IPv4 Relay Statistics for VRF default:

     TYPE          |     RECEIVE     |    TRANSMIT    |      DROP      |
-------------------------------------------------------------------
 DISCOVER          |          2000  |          2000  |            0  |
 OFFER             |          2000  |          2000  |            0  |
 REQUEST           |          5500  |          5500  |            0  |
 DECLINE           |             0  |             0  |            0  |
 ACK               |          5500  |          5500  |            0  |
 NAK               |             0  |             0  |            0  |
 RELEASE           |           500  |           500  |            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 DHCPv4 relay profile details */
Router# show dhcp ivp4 profile name PoA1 relay

Profile: PoA1 relay
Helper Addresses:
        192.0.2.2, vrf default, giaddr 198.51.100.1
Remote-Id Format  : [ascii | hex]
Remote-Id value   : cisco
Information Option: Enabled
Information Option Allow Untrusted: Enabled
Information Option VPN: Enabled
Information Option VPN Mode: RFC
Information Option Policy: Replace
```

Related Topics

- DHCPv4 Relay on IRB, on page 91

Associated Commands

- show dhcp vrf default ipv4 relay statistics
- show dhcp ivp4 profile name

DHCPv4 Relay Synchronization for All-Active Multihoming

DHCPv4 Relay Synchronization for All-active Multihoming feature enables a transitory entity between the end user and DHCPv4 server and does not create any DHCPv4 binding. This feature supports the equal distribution of DHCP control-plane packets among end users across Point of Attachments (PoAs). All DHCP control packets for single users exist on the same DHCPv4 relay (PoA) so that end users can lease IP address allocation without any intervention and delay.

Multiprotocol extension BGP session is established between PEs to edge routers over MPLS-SR so that the learned MAC-IP information is sent over BGP to the edge router. MP-BGP advertises the learned MAC-IP information using route type-2 for a given Ethernet Segment Identifier (ESI) and Ethernet tag. The edge router has the capability of redistributing the routes to other PEs that are learnt from PE1 or PE2, and vice-versa. This mechanism ensures that the MAC-IP routes are distributed to the edge router so that individual PEs have complete MAC-IP routing information.

This feature ensures forwarding of bidirectional traffic. For high availability, during node (PoA#1 or PoA#2) failures, access interface failures, or core link failures, the other PoA forwards data traffic.

DHCPv6 Relay IAPD on IRB

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Relay Identity Association for Prefix Delegation (IAPD) on IRB feature allows the user to manage link, subnet, and site addressing changes. This feature automates the process of assigning prefixes to a customer for use within their network. The prefix delegation occurs between a provider edge (PE) device and customer edge (CE) device using the DHCPv6 prefix delegation option. After the delegated prefixes are assigned to a user, the user may further subnet and assign prefixes to the links in the network.

DHCPv6 relay transmits all request packets that comes over access interface towards external DHCPv6 server to request IAPD (::/64 or ::/48) allocation for the end user. DHCPv6 relay also receives response packets from DHCPv6 server and forwards the packets towards the end users over access interface. DHCPv6 relay acts as stateful for the end users by maintaining DHCPv6 PD binding and respective route entry for the allocated IAPD. DHCPv6 relay supports Internet Assigned Numbers Authority (IANA) and Identity Association for Prefix Delegation (IAPD) address allocation for the end-user. The IAPD prefix is based on prefix-pool that is configured on DHCPv6 server.

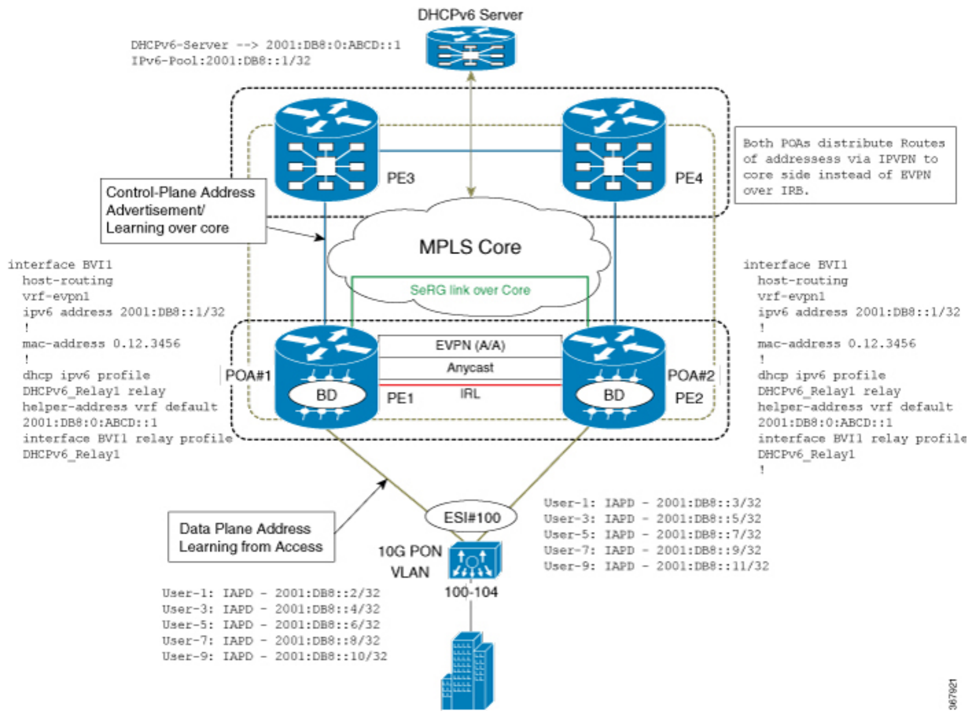
For DHCPv6 relay, access (BVI) interface is tied up with relay profile. Whenever ToRs relay the DHCPv6 packets that are received from client to DHCPv6 server, ToR discovers the best source IP address for a given defined VRF of DHCPv6 server IP address. ToRs maintain unique source IP address for each VRF to reach out DHCPv6 server. DHCPv6 relay has unique IPv4 source IP address defined under loopback interfaces for the defined VRFs of DHCPv6 helper-addresses and routable through MPLS core network.

Anycast IP address configured on the BVI interface acts as a default gateway for end users and address allocation occurs on the same subnet. ToRs maintain unique source IP address to relay DHCPv6 packets towards DHCPv6 server over IPVPN of MPLS core network. The same ToRs receive response packets from external DHCPv6 server. Unique source address on each ToR under DHCPv6 relay is required for DHCPv6 process to maintain the context of packet received over access interface and relayed packet. This mechanism helps to send reply response to end users over BVI interface.

DHPCv6 relay Handling for EVPN and DHCPv6 Server in Default VRF

DHCPv6 relay over EVPN IRB and DHCPv6 servers resides in the same default VRFs. The DHCPv6 relay profiles are associated with helper-addresses of DHCPv6 address under default VRFs. The PEs do not include Relay-Information option in DHPCv6-Relayed packets unlike DHCPv4.

Figure 11: DHPCv6 relay Handling for EVPN and DHCPv6 Server in Default VRF



Configure DHCPv6 Relay IAPD on IRB

Perfrom these tasks to configure DHCPv6 Relay IAPD on IRB.

Configuration Example

```
/* PE1 configuration */

Router# configure
Router(config)# interface BVI1
Router(config-if)# host-routing
Router(config-if)# vrf-evpn1
Router(config-if)# ipv6 address 2001:DB8::1/32
Router(config-if)# exit
Router(config)# mac-address 0.12.3456
!
Router# configure
Router(config)# dhcp ipv6
Router(config-dhcpv6)# profile DHCPv6_Relay1 relay
Router(config-dhcpv6-relay-profile)# helper-address vrf default 2001: DB8:0:ABCD::1
Router(config-dhcpv6-relay-profile)# interface BVI1 relay profile DHCPv6_Relay
Router(config-dhcpv6-relay-profile)# commit

/* PE2 configuration */

Router# configure
Router(config)# interface BVI1
Router(config-if)# host-routing
```

```
Router(config-if)# vrf-evpn1
Router(config-if)# ipv6 address 2001:DB8::1/32
Router(config-if)# exit
Router(config)# mac-address 0.12.3456
!
Router# configure
Router(config)# dhcp ipv6
Router(config-dhcpv6)# profile DHCPv6_Relay1 relay
Router(config-dhcpv6-relay-profile)# helper-address vrf default 2001: DB8:0:ABCD::1
Router(config-dhcpv6-relay-profile)# interface BVI1 relay profile DHCPv6_Relay
Router(config-dhcpv6-relay-profile)# commit
```

Running Configuration

This section shows DHCPv6 Relay IAPD on IRB running configuration.

```
/* PE1 Configuration */
interface BVI1
 host-routing
 vrf-evpn1
 ipv6 address 2001:DB8::1/32
  !
 mac-address 0.12.3456
!
 dhcp ipv6 profile DHCPv6_Relay1 relay
 helper-address vrf default 2001: DB8:0:ABCD::1
 interface BVI1 relay profile DHCPv6_Relay1
!

/* PE2 Configuration */interface BVI1
 host-routing
 vrf-evpn1
 ipv6 address 2001:DB8::1/32
  !
 mac-address 0.12.3456
!
 dhcp ipv6 profile DHCPv6_Relay1 relay
 helper-address vrf default 2001: DB8:0:ABCD::1
 interface BVI1 relay profile DHCPv6_Relay1
!
```

Verification

Verify DHCPv46 Relay IAPD on IRB configuration.

```
/* Verify DHCPv6 relay statistics
Router# show dhcp vrf default ipv6 relay statistics

DHCP IPv6 Relay Statistics for VRF default:

     TYPE          |    RECEIVE    |    TRANSMIT   |      DROP      |
-------------------------------------------------------------------
 DISCOVER          |          2000  |          2000  |             0  |
 OFFER             |          2000  |          2000  |             0  |
 REQUEST           |          5500  |          5500  |             0  |
 DECLINE           |             0  |             0  |             0  |
 ACK               |          5500  |          5500  |             0  |
 NAK               |             0  |             0  |             0  |
 RELEASE           |           500  |           500  |             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  |
```

Related Topics

- DHCPv6 Relay IAPD on IRB, on page 99

Associated Commands

- show dhcp ipv6 relay statistics vrf default

DHCPv6 PD Synchronization for All-Active Multihoming using Session Redundancy

DHCPv6 PD Synchronization for All-Active Multihoming using Session Redundancy feature provides load balancing for both control and data packets. This feature helps in efficient utilization of devices with respect to throughput (line rate) and processing power.

Prior to this release, Session Redundancy (SeRG) mechanism supported active-standby to address access failure, core failure, and node or chassis failures. In all these cases, one active PoA is responsible to create sessions and synchronize binding information using SeRG across the PoA. This mechanism did not serve the purpose of EVPN all-active multihoming as PoAs are in primary-secondary mode for a given access-link in SeRG group. This restricts only one node that acts as primary to process control packets, create bindings, and forward data path.

With DHCPv6 PD Synchronization for All-active Multihoming feature using SeRG group configuration, you can define both POAs to be active unlike in primary-secondary mode. Also, there is no need to exchange or negotiate the roles of respective PoAs.

SeRG does not distribute IAPD prefix routes over BGP in any of the route types. The routed BVI interface is configured with DHCPv6 relay to provide PD allocation for the end user.

Each individual multihoming peer SeRG role is `ACTIVE` only. SeRG does not support any roles other than `NONE` and `ACTIVE`. Define interface-list under SeRG as BVI interface, typically use one or more BVI interfaces. However, it is not recommended to define L2 transport ACs under SeRG interface list because the L2 transport ACs are defined under L2VPN BD, and SeRG-client DHCPv6 is unaware of these AC information.

In SeRG active-active mode, IPv6-ND synchronization is supressed across POAs.

Restrictions

- SeRG does not support core link failures.
- SeRG does not support core and access tracking mechanism.
- Ensure that there are no bindings while configuring `ACTIVE-ACTIVE` mode.
- Ensure that you have the same configuration on all PoAs. The Bundle-Ether L2transport ACs configuration has to be same on both the sides along with BD and BVI configuration.

- **clear session-redundancy** command is not supported in any mode to avoid system inconsistency.
- In SeRG active-active mode, ensure that both PoAs are reachable over core links always. It is recommended to configure EVPN Core Isolation feature, which maps core links to access link. This mechanism ensures to eliminate respective access links whenever core links are down.
- SeRG is supported on the NCS 5500 and NCS 5700 platforms, excluding NCS-57C1, NCS-57B1, and NCS-57D2.

Configure DHCPv6 PD Synchronization

Perfrom these tasks to configure DHCPv6 PD synchronization using SeRG.

Configuration Example

```
/* PoA1 configuration */
Router# configure
Router(config)# session redundancy
Router(config-session-red)# source-interface Loopback0
Router(config-session-red)# group 1
Router(config-session-red-group)# peer 192.0.2.1
Router(config-session-red-group)# mode active-active
Router(config-session-red-group)# interface-list
Router(config-session-red-group-inft)# interface BVI1 id 1
Router(config-session-red-group-intf)# commit

/* PoA2 configuration */
Router# configure
Router(config)# session redundancy
Router(config-session-red)# source-interface Loopback0
Router(config-session-red)# group 1
Router(config-session-red-group)# peer 198.51.100.1
Router(config-session-red-group)# mode active-active
Router(config-session-red-group)# interface-list
Router(config-session-red-group-intf)# interface BVI1 id 1
Router(config-session-red-group-intf)# commit
```

Running Configuration

This section shows DHCPv6 PD synchronization running configuration.

```
/* PoA1 Configuration */
session-redundancy
source-interface Loopback0
group 1
  peer 192.0.2.1
  mode active-active
  interface-list
   interface BVI1 id 1
  !
!
!
/* PoA2 Configuration */
session-redundancy
source-interface Loopback0
group 1
```

```
  peer 198.51.100.1
  mode active-active
  interface-list
   interface BVI1 id 1
  !
!

!
```

Verification

Verify DHCPv6 PD synchronization configuration.

```
/* Verify the session redundancy group */

Router# show session-redundancy group
Wed Nov 28 16:00:36.559 UTC
Session Redundancy Agent Group Summary
Flags      : E - Enabled, D - Disabled, M - Preferred Master, S - Preferred Slave
             H - Hot Mode, W - Warm Mode, T - Object Tracking Enabled
P/S        : Peer Status
             I - Initialize, Y - Retry, X - Cleanup, T - Connecting
             L - Listening, R- Registered, C - Connected, E - Established
I/F-P Count: Interface or Pool Count
SS Count : Session Count
------------------------------------------------------------------------------------------------------------------------

   Node Name    | Group ID | Role | Flags  |          Peer Address         | P/S | I/F-P Count
 |  SS Count   | Sync Pending
------------------------------------------------------------------------------------------------------------------------
0/RP0/CPU0             1  Active   E-H-   120.1.1.1                      E            1
         1               0
0/RP0/CPU0             2  Active   E-H-   120.1.1.1                      E            1
         0               0
0/RP0/CPU0             3  Active   E-H-   120.1.1.1                      E            1
         0               0
0/RP0/CPU0             4  Active   E-H-   120.1.1.1                      E            1
         0               0
0/RP0/CPU0             5  Active   E-H-   120.1.1.1                      E            1
         0               0
------------------------------------------------------------------------------------------------------------------------
Session Summary Count(Master/Slave/Active/Total): 0/0/1/1

/* Verify IPv6 relay binding */

Router# show dhcp ipv6 relay binding
Summary:
Total number of clients: 1

IPv6 Prefix: 60:1:1:1::/64 (BVI1)
    Client DUID: 000100015bfeb921001094000000
    IAID: 0x0
    VRF: default
    Lifetime: 120 secs (00:02:00)
    Expiration: 91 secs (00:01:31)
    L2Intf AC: Bundle-Ether1.1
    SERG State: SERG-ACTIVE
    SERG Intf State: SERG-ACTIVE
```

Related Topics

- DHCPv6 PD Synchronization for All-Active Multihoming using Session Redundancy , on page 102

Associated Commands

- show session-redundancy group
- show dhcp ipv6 relay binding

IAPD Route Distribution and Withdrawal in DHCPv6 Relay

If there is an EVPN Multi-Homing Active-Active scenario, DHCPv6 relay agent is supported over L2VPN bridge domain associated with Attachment Circuits (ACs) and BVI interface with allocation of Identity Association for Prefix Delegation (IAPD) routes. Also, DHCPv6 relay agent performs route distribution using iBGP over the MPLS core network. During core-to-subscriber traffic, few ACs can be down, but BVI is still up because not all ACs are down. This scenario can result in unreported traffic drop for subscribers in ACs that are down. The cause being the IAPD routes that are still intact with the MPLS core network though the ACs are down.

To prevent unreported traffic drop, the DHCPv6 relay agent is enabled to perform IAPD route withdrawal from the MPLS core network over iBGP for sessions. The route withdrawals occur whenever the L2VPN bridge domain ACs are down. Also, whenever the ACs return to the up state, the DHCPv6 relay agent can distribute IAPD routes to the MPLS core network over iBGP.

Layer 3 EVPN IGMP and MLD state synchronization

Layer 3 EVPN IGMP and Multicast Listener Discovery (MLD) state synchronization is a network solution that:

- enables synchronization of IPv4 IGMP and IPv6 MLD states across multiple provider edge (PE) devices,
- supports reliable and seamless delivery of multicast services in residential fiber-to-the-home (FTTH) deployments, and
- eliminates the need for complex Layer 2 (L2) and IRB configurations by using Layer 3 (L3) sub-interfaces.

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
Layer 3 EVPN IGMP and MLD state synchronization	Release 25.4.1	You can ensure seamless and reliable multicast delivery in residential FTTH networks with IGMP and MLD state synchronization for L3 using EVPN. This feature synchronizes IPv4 IGMP and IPv6 Multicast Listener Discovery (MLD) states across multiple PE devices using L3 sub-interfaces, eliminating the need for complex L2 or IRB configurations. It supports both VRF and global routing table deployments, providing flexibility for various network designs.

In many fiber-to-the-home (FTTH) deployments, multicast receivers reside in residential networks where hosts do not communicate with each other. To reduce costs, service providers typically use basic Layer 2 switches in these networks and connect them to PE devices through Layer 2 port channels. However, when PE devices terminate traffic at Layer 2, you must configure Integrated Routing and Bridging (IRB) interfaces to bridge Layer 2 and Layer 3 domains, adding unnecessary complexity given the lack of host to host communication.

Including Layer 2 switching and IRB interfaces in these scenarios increases operational overhead without tangible benefits. To simplify the architecture, you can terminate access connections as Layer 3 subinterfaces on the PE device. This approach removes the need for IRB interfaces, streamlines forwarding logic, and delivers a more efficient and cost-effective design for residential multicast delivery.

IGMP and MLD state synchronization for L3 using EVPN provides consistent IPv4 and IPv6 multicast group membership across multiple PE devices, enabling resilient multicast services in residential FTTH networks by leveraging L3 connectivity. This design is aligned with RFC 9251 for robust and scalable multihoming.

Multihomed topologies for IGMP and MLD state synchronization

Multihomed topologies for IGMP and MLD state synchronization are network designs that

- redundantly connect customer edge (CE) devices to multiple PEs,
- use Layer 3 sub-interfaces for direct connection, and
- support multicast state synchronization to maintain uninterrupted service for both IPv4 and IPv6 multicast traffic.

Benefits of Layer 3 EVPN IGMP and MLD state synchronization

Layer 3 EVPN IGMP and MLD state synchronization provides these benefits.

- Simplifies network architecture by eliminating unnecessary Layer 2 switching in residential deployments.
- Improves scalability and reduces operational overhead by adopting Layer 3 multihoming, as described in RFC 9251.
- Improves multicast performance and reliability by simplifying forwarding, boosting efficiency across IP versions.

Guidelines for Layer 3 EVPN IGMP and MLD state synchronization

General guidelines

- There is no need to change standard multicast routing configuration for L3 EVPN IGMP and MLD state synchronization.
- Only bundle subinterfaces support L3 EVPN IGMP and MLD state synchronization.
- This feature supports both VRF and global routing table deployments.

Protocol recommendations for IGMP and MLD state synchronization

- Configure all IGMP (IPv4) and MLD (IPv6) parameters such as timers, versions, and querier settings, identically across all redundancy groups.
- If static joins are required over multihomed ports, configure them identically across redundancy groups, as static joins are not automatically synchronized.

Redundancy mode support for IGMP and MLD state synchronization

When configuring PIM/PIMv6 with IGMP or MLD state synchronization, use only all-active redundancy mode for multicast. Single-active and port-active modes are not supported.

How Layer 3 EVPN IGMP and MLD state synchronization works

Summary

The key components involved in IGMP and MLD state synchronization are:

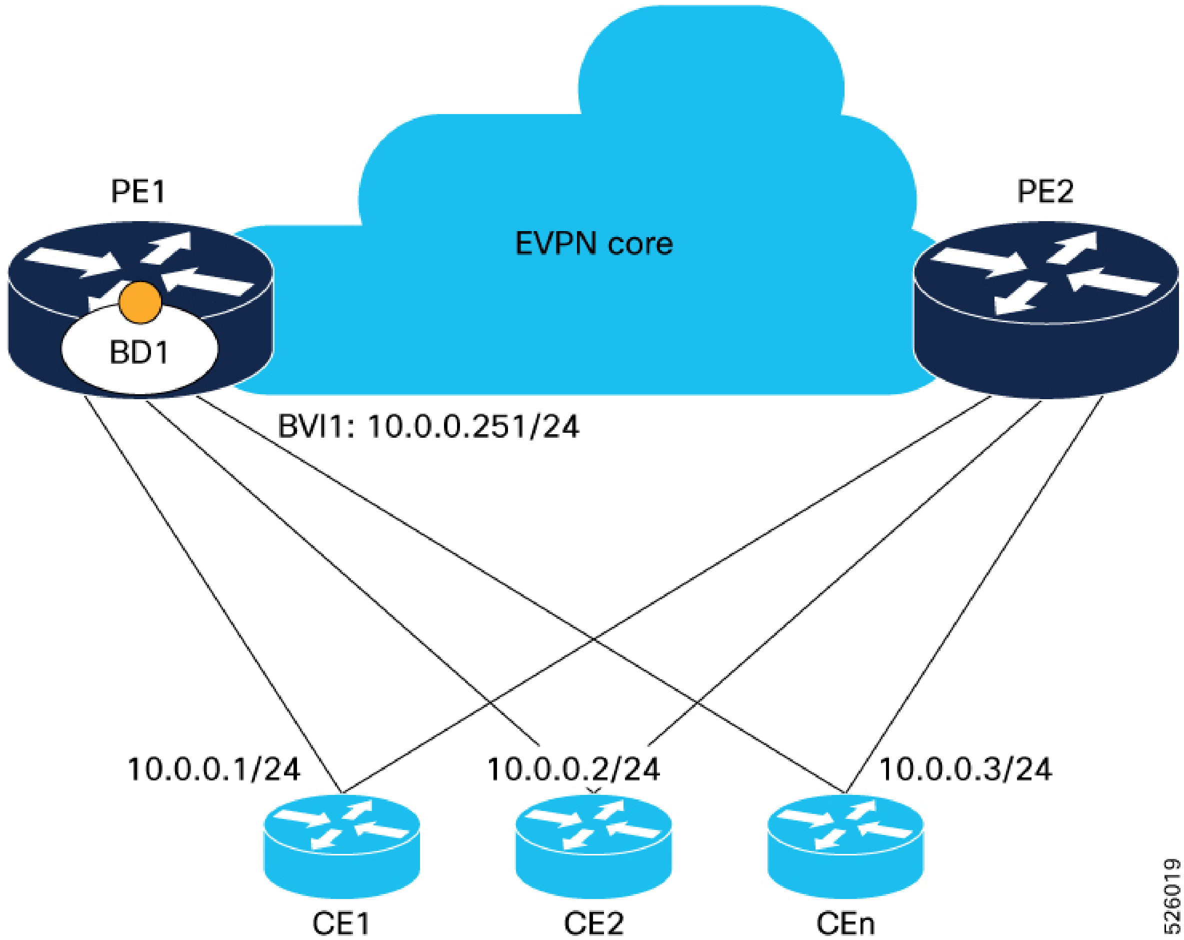
- Layer 3 (L3)
    - PE1 and PE2: Both act as PE routers running PIM/PIMv6 (Protocol Independent Multicast for IPv4/IPv6) and connect to the customer edge (CE) via Bundle Ethernet interfaces.
    - Global routing table (GRT): Handles unicast and multicast routing.
    - Designated router (DR): Manages multicast group membership.
    - Designated forwarder (DF): Forwards multicast traffic on the shared segment as elected per EVPN rules.

      In EVPN multihoming scenarios, the DF is responsible for forwarding multicast traffic, including IGMP and MLD queries, to the CE. Only the DF PE forwards such traffic, while the Non-Designated Forwarder (NDF) blocks it to prevent duplication and loops. This mechanism ensures that the CE receives a single, loop-free copy of multicast queries and traffic.
    - EVPN A-A (All-active): Enables both PEs to be active and participate in forwarding.
- Layer 2 (L2)
    - CE: Device that receives L2 multicast (IGMP/MLD) joins from FTTH subscribers.

IGMP and MLD state synchronization ensures efficient and loop-free multicast delivery in EVPN active-active multihoming environments by coordinating state and forwarding roles among Layer 2 and Layer 3 network components.

Workflow

Figure 12: A sample toplogy for L3 EVPN IGMP and MLD state synchronization



This diagram illustrates the operation of L3 EVPN for synchronizing IGMP and MLD state across PE devices in a dual-homed all-active topology. The goal is to ensure consistent multicast group state between PE1 and PE2 for seamless multicast forwarding, even in the event of a link or device failure.

This example focuses on multicast running in the global routing table (GRT), but the process is the same when multicast operates in a VRF (MVPN).

These stages describe multicast state synchronization:

1. Reception of IGMP/MLD joins from the customer edge (CE):

   The CE device sends IGMP (for IPv4) or MLD (for IPv6) membership reports upstream to either PE1 or PE2 over the Layer 2 Bundle Ethernet interface to indicate interest in specific multicast groups.

2. State learning and actions:

   The PE that receives the IGMP or MLD join sends a PIM or PIMv6 join message to the core to request the multicast stream. Because both PEs act as PIM or PIMv6 DRs for the Bundle-Ether interface, either PE that receives the IGMP or MLD join will trigger a PIM or PIMv6 join to the core.

3. State synchronization between PEs:

IGMP and MLD state information is synchronized between PE1 and PE2 using EVPN mechanisms. The PEs exchange synchronization messages containing IGMP and MLD states, including group membershipand source details, over the EVPN all-active control plane. This ensures both PEs maintain an identical view of multicast group membership, regardless of which PE received the original join.

4. Live-live redundancy:

   IGMP/MLD state synchronization allows the second PE to also send PIM or PIMv6 join towards the core to request the multicast stream. Both PEs receive the multicast stream from the core, ensuring live-live redundancy.

5. Multicast traffic forwarding:

   Both PEs receive the multicast traffic, but only the DF sends it through the Bundle-Ether interface. The non-designated forwarder (NDF) does not forward the traffic.

6. Failure scenario:

   If the Bundle-Ether interface on the DF fails, the second PE automatically becomes the DF and immediately forwards the multicast stream. This ensures uninterrupted multicast delivery.

Example scenario

1. A subscriber sends an IGMP join message for a TV multicast group from behind the CE device.
2. The CE forwards the IGMP join on one of the two bundle links; for example, the join is received by PE1.
3. Both PE1 and PE2 are designated routers (DRs). Regardless of which PE receives the IGMP join, a Protocol Independent Multicast (PIM) join is sent upstream to request multicast traffic.
4. PE1 exchanges its IGMP state with PE2 over EVPN active-active synchronization.
5. When PE2 receives the IGMP state over EVPN A-A, it also sends a PIM join upstream.
6. PE1 forwards the multicast stream through the Bundle-Ether interface because it is the DF.
7. If PE1 fails, PE2 already maintains the multicast group state and receives the stream from the core, allowing it to immediately take over forwarding multicast traffic to the subscriber.

Configure Layer 3 EVPN IGMP and MLD state synchronization

Before you begin

- Enable and activate the **l2vpn evpn** address family on peering with route reflectors to allow EVPN control plane messages to be exchanged between PEs.
- Ensure that the same ESI value is configured under the EVPN settings on both PEs for the Bundle-Ether interface.

  For more details on EVPN configuration, see EVPN Single-Active Multihoming for Anycast Gateway IRB.
- Layer 3 EVPN IGMP and MLD state synchronization requires the standard multicast configuration. For more details, see the *Multicast Configuration Guide for Cisco NCS 5500 Series Routers*.

Procedure

Step 1 Configure the Ethernet Virtual Instance (EVI) to be used for EVPN synchronization.

a) Run the **evpn route-sync <evi>** command if multicast operates in GRT.

Example:

```
Router#config
Router(config)#evpn route-sync 10
Router(config-evpn-instance)#commit
```

The **evpn route-sync <evi>** command enables route synchronization for a specific EVI in the GRT. Use this command when multicast operates in the GRT and the ESI and associated bundle subinterfaces are not tied to any VRF. The command ensures that Layer 3 routes for the EVPN instance are synchronized across all PE devices within the GRT.

b) Run the **evpn route-sync <evi>** command if multicast operates in default VRF.

Example:

```
Router#config
Router(config)#evpn
Router(config-evpn)#route-sync 10
Router(config-evpn-instance)#vrf default
Router(config-evpn-instance)#exit
Router(config-evpn)#interface Bundle-Ether1
Router(config-evpn-ac)#ethernet-segment
Router(config-evpn-ac-es)#identifier type 0 00.01.00.ac.00.00.01.0a.00
Router(config-evpn-ac-es)#commit
```

The **evpn route-sync <evi>** command enables route synchronization for a specific EVI in the default VRF context. This command synchronizes Layer 3 routes learned on bundle subinterfaces or Ethernet segments for that EVPN instance across all PE devices within the default VRF.

Note
The ethernet-segment **identifier type** must match the one configured on the remote dual-homed PE router.

c) Run the **vrf <name> evpn-route-sync <evi>** command if multicast operates in a private VRF.

Example:

```
Router#config
Router(config)#evpn
Router(config-evpn)#interface Bundle-Ether1
Router(config-evpn-ac)#ethernet-segment
Router(config-evpn-ac-es)#identifier type 0 00.01.00.ac.00.00.01.0a.00
Router(config-evpn-ac-es)#root
Router(config)#vrf vrf-name evpn-route-sync 10
Router(config)#commit
```

The **vrf <name> evpn-route-sync <evi>** command enables route synchronization for a specific EVI within a non-default VRF context. Use this command when the Ethernet segment identifier (ESI) and related bundle subinterfaces are part of a non-default VRF. The command ensures that Layer 3 routes for that EVPN instance are synchronized across all PE devices in the specified VRF.

Use a unique EVI number for this CLI; do not use the same EVI under **l2vpn** configuration for standard EVPN services.

Step 2 Run the **show mfib platform evpn bucket loc <location>** command to display EVPN bucket information for the specified location.

Example:

```
Router#show mfib platform evpn bucket loc 0/0/CPU0

-------------------------------------------------------------------------------
ESI Interface        Handle     Bucket ID  State  Ref cnt    Stale  Del pending
-------------------------------------------------------------------------------
BE1                  0x7800008c 8          NDF    0          F      F
BE1                  0x7800008c 9          DF     0          F      F
BE1                  0x7800008c 10         NDF    0          F      F
BE1                  0x7800008c 11         DF     0          F      F
BE1                  0x7800008c 4          NDF    0          F      F
BE1                  0x7800008c 5          DF     4          F      F
BE1                  0x7800008c 6          NDF    0          F      F
BE1                  0x7800008c 7          DF     0          F      F
BE1                  0x7800008c 0          NDF    0          F      F
BE1                  0x7800008c 1          DF     0          F      F
BE1                  0x7800008c 2          NDF    0          F      F
BE1                  0x7800008c 3          DF     0          F      F
BE2                  0x78000094 4          NDF    0          F      F
BE2                  0x78000094 5          DF     0          F      F
BE2                  0x78000094 6          NDF    0          F      F
BE2                  0x78000094 7          DF     0          F      F
BE2                  0x78000094 0          NDF    0          F      F
BE2                  0x78000094 1          DF     0          F      F
BE2                  0x78000094 2          NDF    0          F      F
BE2                  0x78000094 3          DF     0          F      F
BE2                  0x78000094 8          NDF    0          F      F
BE2                  0x78000094 9          DF     0          F      F
BE2                  0x78000094 10         NDF    0          F      F
BE2                  0x78000094 11         DF     0          F      F
-------------------------------------------------------------------------------
Router#
```

This output shows that EVPN multi-homing is enabled with multiple buckets per ESI, and DF roles are distributed across different buckets and interfaces. The mix of DF and NDF states indicates active load balancing and redundancy, ensuring resiliency in multicast forwarding. No entries are stale or pending deletion.

Step 3 Run the **show mrib platform idb** command to display the multicast routing interface database (IDB) information for all interfaces on the platform.

Example:

```
Router#show mrib platform idb

-------------------------------------------------------
IDB Hash Table (Total Count 5)
-------------------------------------------------------
  -------------------------------------------------------------
  Bundle-Ether1 (0x7800008c)
  -------------------------------------------------------------
  -------------------------------------------------------------

  -------------------------------------------------------------
  Bundle-Ether2 (0x78000094)
  -------------------------------------------------------------
  -------------------------------------------------------------

  -------------------------------------------------------------
  Bundle-Ether1.10 (0x7800009c)
  -------------------------------------------------------------
```

```
  Root Interface:     Bundle-Ether1 (0x7800008c)
  EVPN registered:    T
  ESI IFH:            0x7800008c
  MH count:           2
 --------------------------------------------------------------

 --------------------------------------------------------------
 Bundle-Ether1.11 (0x780000a4)
 --------------------------------------------------------------
  Root Interface:     Bundle-Ether1 (0x7800008c)
  EVPN registered:    T
  ESI IFH:            0x7800008c
  MH count:           2
 --------------------------------------------------------------

 --------------------------------------------------------------
 Bundle-Ether2.10 (0x780000d4)
 --------------------------------------------------------------
  Root Interface:     Bundle-Ether2 (0x78000094)
  EVPN registered:    T
  ESI IFH:            0x78000094
  MH count:           2
 --------------------------------------------------------------

-------------------------------------------------------
```

This output shows that sub-interfaces Bundle-Ether1.10, Bundle-Ether1.11, and Bundle-Ether2.10 are registered for EVPN multi-homing, each associated with their root bundle and ESI. The MH count indicates that each sub-interface is participating in a multi-homing group with two members, confirming redundancy and active EVPN multi-homing configuration.

Step 4 Run the **show mfib hardware route** command to display multicast forwarding details, highlighting EVPN multi-homing information for each outgoing interface.

a) On Designated Forwarder (DF) node:

Example:

```
Router# show mfib hardware route 50.0.0.2 232.1.2.3 detail location 0/RP0/CPU0
Wed Oct  8 16:11:06.257 UTC
Route (50.0.0.2: 232.1.2.3)
   HAL PD context
       VRF ID: 0 Core MCID : 0 Core backup MCID 0 Routeid: 4 trans_id 121

   HAL Ingress route context:
       Route FGID: 9220 RPF IF signal : not-set Local receivers: set
       Encap ID flag: not-set, Encap ID: 0
       Tunnel RIF: 0x0
       Statistics enabled: not-set

       Ingress engine context:
          local_route: set, is_accept_intf_bvi: not-set is_tun_rif_set:not-set
           VRF ID: 0 RPF ID:0 Tunnel RIF:0x0
   HAL Egress route context:
       RPF ID: 0

       Egress engine context:
          bvi_count: 0

   DPA Route context:
      Handle: 308beee348
      Dpa_trans_id: 4308
      Number of OLE: 2   VRF ID: 0
      Incoming interface : Te0/0/0/6 A_intf_id: 0x11 Merged flag 0
```

```
  Tunnel RIF : 0x0 Tunnel LIF : 0x0 FGID: 9220
  FEC ID : 0x20013372 Punt action: 0x0
  TCAM entry ID : 0x0 IPMC action: 0x0 FEC Accessed 1
  L3 Intf Refhandle : 0x30920fe1b8 L3 interface ref key: 0x0
  Statistics enabled : not-set Statistics activated : not-set

  NPU Ingress Resource:
    NPU: 0
    vrf:0 ,src:50.0.0.2, grp:232.1.2.3, result: 0x20013372
    FEC Idx: 0x20013372, RPF: 17 MCID: 9220

  Egress Route OLEs:
    Handle: 30930e54f8
    Dpa_trans_id: 4335
    NPU ID: 0 Outgoing intf: Te0/0/0/12
    OLE Type : Bundle Sub-interface
    is_recycle_lif_ole: not-set recycle_lif:0
    outgoing port : 0x1f cud: 0x327048 is_bundle: 1
    Sys_port : 0x0 mpls encap id: 0x0 LAG ID: 0
    is_pw_access: 0 pw_encap_id:0
    L3 intf refhndl : 0x30921037d8 L3 intf refkey: 0x3c00801c
    L2 Port refhandle : 0x3092113838 L2 Port refkey: 0x3c0000f8
    MPLS nh refhandle : 0x0 MPLS nh refkey: 0x0
    LAG port refhandle : 0x3092a040e8 LAG port refkey: 0x3c008014
    EFP-Visibility: not-set
    Part of Bundle Interface BE1.10, Local N Sysport 0x0
    Total fwd packets : 0 Total fwd bytes: 0


EVPN Multi-homing Information
-------------------------------------
Intf            ESI IFH    Bucket ID Designated Forwarder
-------------------------------------
BE1.10          0x3c008014 4   DF
-------------------------------------
    NPU ID: 0 Outgoing intf: Te0/0/0/12
    OLE Type : Bundle Sub-interface
    is_recycle_lif_ole: not-set recycle_lif:0
    outgoing port : 0x1f cud: 0x32704a is_bundle: 1
    Sys_port : 0x0 mpls encap id: 0x0 LAG ID: 0
    is_pw_access: 0 pw_encap_id:0
    L3 intf refhndl : 0x3092103488 L3 intf refkey: 0x3c008024
    L2 Port refhandle : 0x3092113838 L2 Port refkey: 0x3c0000f8
    MPLS nh refhandle : 0x0 MPLS nh refkey: 0x0
    LAG port refhandle : 0x3092a040e8 LAG port refkey: 0x3c008014
    EFP-Visibility: not-set
    Part of Bundle Interface BE1.20, Local N Sysport 0x0
    Total fwd packets : 0 Total fwd bytes: 0


EVPN Multi-homing Information
-------------------------------------
Intf            ESI IFH    Bucket ID Designated Forwarder
-------------------------------------
BE1.20          0x3c008014 4   DF
-------------------------------------
  NPU Egress Resource:
    NPU: 0
    replications: 2
    gport: 0x400001f cud1: 0x327048 cud2: 0xffffffff
    gport: 0x400001f cud1: 0x32704a cud2: 0xffffffff
```

b) On Non-Designated Forwarder (NDF) node:

Example:

```
Router# show mfib hardware route 50.0.0.2 232.1.2.3 detail location 0/0/CPU0
Route (50.0.0.2: 232.1.2.3)
   HAL PD context
       VRF ID: 0 Core MCID : 0 Core backup MCID 0 Routeid: 16 trans_id 459

   HAL Ingress route context:
       Route FGID: 8203 RPF IF signal : not-set Local receivers: set
       Encap ID flag: not-set, Encap ID: 0
       Tunnel RIF: 0x0
       Statistics enabled: not-set

       Ingress engine context:
          local_route: set, is_accept_intf_bvi: not-set is_tun_rif_set:not-set
           VRF ID: 0 RPF ID:0 Tunnel RIF:0x0
   HAL Egress route context:
       RPF ID: 0

       Egress engine context:
          bvi_count: 0

   DPA Route context:
      Handle: 308cc9d348
      Dpa_trans_id: 90083
      Number of OLE: 2   VRF ID: 0
      Incoming interface : Te0/0/0/20 A_intf_id: 0xb Merged flag 0
      Tunnel RIF : 0x0 Tunnel LIF : 0x0 FGID: 8203
      FEC ID : 0x2000ffb5 Punt action: 0x0
      TCAM entry ID : 0x0 IPMC action: 0x4 FEC Accessed 1
      L3 Intf Refhandle : 0x308f854dd8 L3 interface ref key: 0x0
      Statistics enabled : not-set Statistics activated : not-set

      NPU Ingress Resource:
        NPU: 0
        vrf:0 ,src:50.0.0.2, grp:232.1.2.3, result: 0x2000ffb5
        FEC Idx: 0x2000ffb5, RPF: 11 MCID: 8203

      Egress Route OLEs:
        Handle: 30903ed678
        Dpa_trans_id: 90102
        NPU ID: 0 Outgoing intf: Te0/0/0/2
        OLE Type : Bundle Sub-interface
        is_recycle_lif_ole: not-set recycle_lif:0
        outgoing port : 0x14 cud: 0x0 is_bundle: 1
        Sys_port : 0x0 mpls encap id: 0x0 LAG ID: 0
        is_pw_access: 0 pw_encap_id:0
        L3 intf refhndl : 0x308f85b7d8 L3 intf refkey: 0x20008024
        L2 Port refhandle : 0x308f86c918 L2 Port refkey: 0xa0
        MPLS nh refhandle : 0x0 MPLS nh refkey: 0x0
        LAG port refhandle : 0x308fd3c0e8 LAG port refkey: 0x20008014
        EFP-Visibility: not-set
        Part of Bundle Interface BE1.20, Local N Sysport 0x0
        Total fwd packets : 0 Total fwd bytes: 0


    EVPN Multi-homing Information
    -------------------------------------
    Intf            ESI IFH    Bucket ID Designated Forwarder
    -------------------------------------
    BE1.20          0x20008014 4   NDF
    -------------------------------------
        NPU ID: 0 Outgoing intf: Te0/0/0/2
        OLE Type : Bundle Sub-interface
```

```
    is_recycle_lif_ole: not-set recycle_lif:0
    outgoing port : 0x14 cud: 0x0 is_bundle: 1
    Sys_port : 0x0 mpls encap id: 0x0 LAG ID: 0
    is_pw_access: 0 pw_encap_id:0
    L3 intf refhndl : 0x308f85b488 L3 intf refkey: 0x2000801c
    L2 Port refhandle : 0x308f86c918 L2 Port refkey: 0xa0
    MPLS nh refhandle : 0x0 MPLS nh refkey: 0x0
    LAG port refhandle : 0x308fd3c0e8 LAG port refkey: 0x20008014
    EFP-Visibility: not-set
    Part of Bundle Interface BE1.10, Local N Sysport 0x0
    Total fwd packets : 0 Total fwd bytes: 0


EVPN Multi-homing Information
-------------------------------------
Intf            ESI IFH    Bucket ID Designated Forwarder
-------------------------------------
BE1.10          0x20008014 4   NDF
-------------------------------------
  NPU Egress Resource:
    NPU: 0
    replications: 2
    gport: 0x4000014 cud1: 0x2b cud2: 0x11846
    gport: 0x4000014 cud1: 0x2c cud2: 0x11848
```