



EVPN Features

This chapter describes how to configure Layer 2 Ethernet VPN (EVPN) features on the router.

- [EVPN Overview, on page 1](#)
- [EVPN Concepts, on page 2](#)
- [EVPN Operation, on page 3](#)
- [EVPN Route Types, on page 5](#)
- [EVPN Timers, on page 6](#)
- [Configure EVPN L2 Bridging Service, on page 8](#)
- [EVPN Modes, on page 10](#)
- [EVPN MPLS Seamless Integration with VPLS, on page 15](#)
- [Configure EVPN on the Existing VPLS Network, on page 16](#)
- [EVPN Features, on page 24](#)
- [CFM Support for EVPN, on page 80](#)
- [CFM on EVPN ELAN, on page 81](#)
- [EVPN Routing Policy, on page 84](#)

EVPN Overview

Ethernet VPN (EVPN) is a solution that provides Ethernet multipoint services over MPLS networks. EVPN operates in contrast to the existing Virtual Private LAN Service (VPLS) by enabling control-plane based MAC learning in the core. In EVPN, PEs participating in the EVPN instances learn customer MAC routes in control-plane using MP-BGP protocol. Control-plane MAC learning brings a number of benefits that allow EVPN to address the VPLS shortcomings, including support for multihoming with per-flow load balancing.

EVPN provides the solution for network operators for the following emerging needs in their network:

- Data center interconnect operation (DCI)
- Cloud and services virtualization
- Remove protocols and network simplification
- Integration of L2 and L3 services over the same VPN
- Flexible service and workload placement
- Multi-tenancy with L2 and L3 VPN

- Optimal forwarding and workload mobility
- Fast convergence
- Efficient bandwidth utilization

EVPN Benefits

The EVPN provides the following benefits:

- **Integrated Services:** Integrated L2 and L3 VPN services, L3VPN-like principles and operational experience for scalability and control, all-active multihoming and PE load-balancing using ECMP, and enables load balancing of traffic to and from CEs that are multihomed to multiple PEs.
- **Network Efficiency:** Eliminates flood and learn mechanism, fast-reroute, resiliency, and faster reconvergence when the link to dual-homed server fails, optimized Broadcast, Unknown-unicast, Multicast (BUM) traffic delivery.
- **Service Flexibility:** MPLS data plane encapsulation, support existing and new services types (E-LAN, E-Line), peer PE auto-discovery, and redundancy group auto-sensing.

EVPN Modes

The following EVPN modes are supported:

- **Single-homing** - Enables you to connect a customer edge (CE) device to one provider edge (PE) device.
- **Multihoming** - Enables you to connect a customer edge (CE) device to more than one provider edge (PE) device. Multihoming ensures redundant connectivity. The redundant PE device ensures that there is no traffic disruption when there is a network failure. Following are the types of multihoming:
 - **Single-Active** - In single-active mode only a single PE among a group of PEs attached to the particular Ethernet-Segment is allowed to forward traffic to and from that Ethernet Segment.
 - **All-Active** - In all-active mode all the PEs attached to the particular Ethernet-Segment is allowed to forward traffic to and from that Ethernet Segment.

EVPN Concepts

To implement EVPN features, you need to understand the following concepts:

- **Ethernet Segment (ES):** An Ethernet segment is a set of Ethernet links that connects a multihomed device. If a multi-homed device or network is connected to two or more PEs through a set of Ethernet links, then that set of links is referred to as an Ethernet segment. The Ethernet segment route is also referred to as Route Type 4. This route is used for designated forwarder (DF) election for BUM traffic.
- **Ethernet Segment Identifier (ESI):** Ethernet segments are assigned a unique non-zero identifier, which is called an Ethernet Segment Identifier (ESI). ESI represents each Ethernet segment uniquely across the network.
- **EVI:** The EVPN instance (EVI) is represented by the virtual network identifier (VNI). An EVI represents a VPN on a PE router. It serves the same role of an IP VPN Routing and Forwarding (VRF), and EVIs are assigned import/export Route Targets (RTs). Depending on the service multiplexing behaviors at the

User to Network Interface (UNI), all traffic on a port (all-to-one bundling), or traffic on a VLAN (one-to-one mapping), or traffic on a list/range of VLANs (selective bundling) can be mapped to a Bridge Domain (BD). This BD is then associated to an EVI for forwarding towards the MPLS core.

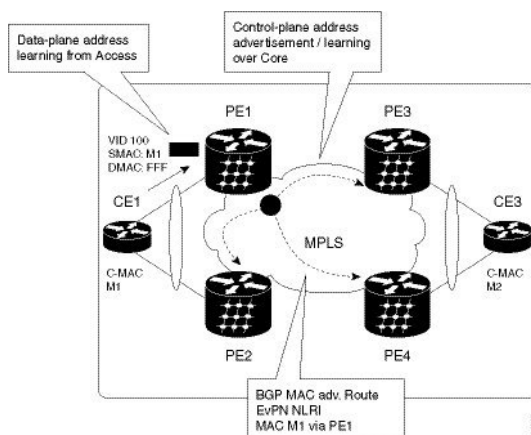
- **EAD/ES:** Ethernet Auto Discovery Route per ES is also referred to as Route Type 1. This route is used to converge the traffic faster during access failure scenarios. This route has Ethernet Tag of 0xFFFFFFFF.
- **EAD/EVI:** Ethernet Auto Discovery Route per EVI is also referred to as Route Type 1. This route is used for aliasing and load balancing when the traffic only hashes to one of the switches. This route cannot have Ethernet tag value of 0xFFFFFFFF to differentiate it from the EAD/ES route.
- **Aliasing:** It is used for load balancing the traffic to all the connected switches for a given Ethernet segment using the Route Type 1 EAD/EVI route. This is done irrespective of the switch where the hosts are actually learned.
- **Mass Withdrawal:** It is used for fast convergence during the access failure scenarios using the Route Type 1 EAD/ES route.
- **DF Election:** It is used to prevent forwarding of the loops. Only a single router is allowed to decapsulate and forward the traffic for a given Ethernet Segment.

EVPN Operation

At startup, PEs exchange EVPN routes in order to advertise the following:

- **VPN membership:** The PE discovers all remote PE members of a given EVI. In the case of a multicast ingress replication model, this information is used to build the PEs flood list associated with an EVI. BUM labels and unicast labels are exchanged when MAC addresses are learned.
- **Ethernet segment reachability:** In multihoming scenarios, the PE auto-discovers remote PE and their corresponding redundancy mode (all-active or single-active). In case of segment failures, PEs withdraw the routes used at this stage in order to trigger fast convergence by signaling a MAC mass withdrawal on remote PEs.
- **Redundancy Group membership:** PEs connected to the same Ethernet segment (multihoming) automatically discover each other and elect a Designated Forwarder (DF) that is responsible for forwarding Broadcast, Unknown unicast and Multicast (BUM) traffic for a given EVI.

Figure 1: EVPN Operation



EVPN can operate in single-homing or dual-homing mode. Consider single-homing scenario, when EVPN is enabled on PE, Route Type 3 is advertised where each PE discovers all other member PEs for a given EVPN instance. When an unknown unicast (or BUM) MAC is received on the PE, it is advertised as EVPN Route Type 2 to other PEs. MAC routes are advertised to the other PEs using EVPN Route Type 2. In multihoming scenarios, Route Types 1, 3, and 4 are advertised to discover other PEs and their redundancy modes (single-active or all-active). Use of Route Type 1 is to auto-discover other PE which hosts the same CE. The other use of this route type is to fast route unicast traffic away from a broken link between CE and PE. Route Type 4 is used for electing designated forwarder. For instance, consider the topology when customer traffic arrives at the PE, EVPN MAC advertisement routes distribute reachability information over the core for each customer MAC address learned on local Ethernet segments. Each EVPN MAC route announces the customer MAC address and the Ethernet segment associated with the port where the MAC was learned from and its associated MPLS label. This EVPN MPLS label is used later by remote PEs when sending traffic destined to the advertised MAC address.

Behavior Change due to ESI Label Assignment

To adhere to RFC 7432 recommendations, the encoding or decoding of MPLS label is modified for extended community. Earlier, the lower 20 bits of extended community were used to encode the split-horizon group (SHG) label. Now, the SHG label encoding uses from higher 20 bits of extended community.

According to this change, routers in same ethernet-segment running old and new software release versions decodes extended community differently. This change causes inconsistent SHG labels on peering EVPN PE routers. Almost always, the router drops BUM packets with incorrect SHG label. However, in certain conditions, it may cause remote PE to accept such packets and forward to CE potentially causing a loop. One such instance is when label incorrectly read as NULL.

To overcome this problem, Cisco recommends you to:

- Minimize the time both PEs are running different software release versions.
- Before upgrading to a new release, isolate the upgraded node and shutdown the corresponding AC bundle.
- After upgrading both the PEs to the same release, you can bring both into service.

Similar recommendations are applicable to peering PEs with different vendors with SHG label assignment that does not adhere to RFC 7432.

EVPN Route Types

The EVPN network layer reachability information (NLRI) provides different route types.

Table 1: EVPN Route Types

Route Type	Name	Usage
1	Ethernet Auto-Discovery (AD) Route	Few routes are sent per ES, carries the list of EVIs that belong to ES
2	MAC/IP Advertisement Route	Advertise MAC, address reachability, advertise IP/MAC binding
3	Inclusive Multicast Ethernet Tag Route	Multicast Tunnel End point discovery
4	Ethernet Segment Route	Redundancy group discovery, DF election
5	IP Prefix Route	Advertise IP prefixes.

Route Type 1: Ethernet Auto-Discovery (AD) Route

The Ethernet Auto-Discovery (AD) routes are advertised on per EVI and per ESI basis. These routes are sent per ES. They carry the list of EVIs that belong to the ES. The ESI field is set to zero when a CE is single-homed. This route type is used for mass withdrawal of MAC addresses and aliasing for load balancing.

Route Type 2: MAC/IP Advertisement Route

These routes are per-VLAN routes, so only PEs that are part of a VNI require these routes. The host's IP and MAC addresses are advertised to the peers within NLRI. The control plane learning of MAC addresses reduces unknown unicast flooding.

Route Type 3: Inclusive Multicast Ethernet Tag Route

This route establishes the connection for broadcast, unknown unicast, and multicast (BUM) traffic from a source PE to a remote PE. This route is advertised on per VLAN and per ESI basis.

Route Type 4: Ethernet Segment Route

Ethernet segment routes enable to connect a CE device to two or PE devices. ES route enables the discovery of connected PE devices that are connected to the same Ethernet segment.

Route Type 5: IP Prefix Route

The IP prefixes are advertised independently of the MAC-advertised routes. With EVPN IRB, host route /32 is advertised using RT-2 and subnet /24 is advertised using RT-5.



Note With EVPN IRB, host route /32 are advertised using RT-2 and subnet /24 are advertised using RT-5.

EVPN Timers

The following table shows various EVPN timers:

Table 2: EVPN Timers

Timer	Range	Default Value	Trigger	Applicability	Action	Sequence
startup-cost-in	30-86400s	disabled	node recovered*	Single-Homed, All-Active, Single-Active	Postpone EVPN startup procedure and Hold AC link(s) down to prevent CE to PE forwarding. Startup-cost-in timer allows PE to set core protocols first.	1
peering	0-3600s	3s	node recovered, interface recovered	All-Active, Single-Active	Starts after sending EVPN RT4 to postpone rest of EVPN startup procedure. Peering timer allows remote PE (multihoming AC with same ESI) to process RT4 before DF election will happen.	3

Timer	Range	Default Value	Trigger	Applicability	Action	Sequence
global mac evpn timer	0-300s	300s	when BGP is fired	Single-Flow-Active and Multi homed all active	Delay the time and effort required to delete the remote portion to save programming cycles working for forwarding path first.	4



- Note**
- The timers are available in EVPN global configuration mode and in EVPN interface sub-configuration mode.
 - Startup-cost-in is available in EVPN global configuration mode only.
 - Timers are triggered in sequence (if applicable).
 - Cost-out in EVPN global configuration mode brings down AC link(s) to prepare node for reload or software upgrade.

* indicates all required software components are loaded.

** indicates link status is up.

*** you can change the recovery timer on Single-Homed AC if you do not expect any STP protocol convergence on connected CE.

Global MAC EVPN Timer

Global mac evpn timer is configurable under **evpn timers mac-postpone** timer. Global MAC EVPN timer is relevant for SYNC routes only in the following scenarios:

- FRR (fast re-route) is configured: MAC and MAC+IP deletes are postponed to help with convergence.
- All-active: MAC+IPs deletes are postponed to allow time for ARP to converge.
- Single-flow-active: MAC+IP deletes are postponed to allow speculative (Address Resolution Protocol) ARP to point to local adjacency.

Typically, a route that is deleted is always quickly learned locally. Using this knowledge, we can delay the time and effort required to delete the remote portion to save programming cycles working for forwarding path first.



- Note** The timer of 5-minutes start when EVPN receives a delete from BGP. The timer doesn't start at the exact time of AC shut or mass-withdraw.

The benefit of this speculative behavior is that we can reduce MAC-IP delete/re-create churn in forwarding and BGP.

Triggers of Global Mac EVPN Timer:

- Shut / No shut on IRB/BVI Interfaces.
- Removing and adding AC Interface Configuration.
- Removing and adding BVI Interface Configuration.
- Removing and adding BVI Interface from Bridge Domains.
- Shut / No shut on AC/Main-port Interface Configuration.

Configure EVPN L2 Bridging Service

Perform the following steps to configure EVPN L2 bridging service.



Note Always ensure to change the label mode from per-prefix to per-VRF label mode. Since L2FIB and VPNv4 route (labels) shares the same resource, BVI ping fails when you exhaust the resources.



Note Traffic to directly connected neighbor on EVPN or VPLS bridge won't work in the following scenarios:

- If neighbor doesn't advertise MPLS explicit null.
- If imposition node has a mix of implicit-null and labeled paths in ECMP or LFA deployment.



Note A device can contain up to 128K MAC address entries. A bridge domain on a device can contain up to 64K MAC address entries.



Note Flooding disable isn't supported on EVPN bridge domains.

```

/* Configure address family session in BGP */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router#(config)# router bgp 200
RP/0/RSP0/CPU0:router#(config-bgp)# bgp router-id 209.165.200.227
RP/0/RSP0/CPU0:router#(config-bgp)# address-family l2vpn evpn
RP/0/RSP0/CPU0:router#(config-bgp)# neighbor 10.10.10.10
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# remote-as 200
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# description MPLSFACING-PEER
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# update-source Loopback 0
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# address-family l2vpn evpn

/* Configure EVI and define the corresponding BGP route targets */

```




Note EVI route target used for multicast EVPN supports only extcomm type sub-type 0xA for EVI route target, the two-octet Autonomous System (AS) specific Extended Community. This means that when using a 4-byte AS number for BGP, you must additionally configure BGP import and export route targets under the EVPN configuration.

```
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)# exit
Router(config-evpn-evi)# advertise-mac

/* Configure a bridge domain */
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
Router(config-l2vpn-bg-bd-ac)# evi 6005
Router(config-l2vpn-bg-bd-ac-evi)# commit
Router(config-l2vpnbg-bd-ac-evi)# exit
```

Running Configuration

```
router bgp 200 bgp
router-id 209.165.200.227
address-family l2vpn evpn
neighbor 10.10.10.10
  remote-as 200 description MPLS-FACING-PEER
  updatesource Loopback0
  addressfamily l2vpn evpn
!

configure
evpn
evi 6005
  bgp
  rd 200:50
  route-target import 100:6005
  route-target export 100:6005
!
advertise-mac

configure
l2vpn
bridge group 1
  bridge-domain 1-1
  interface GigabitEthernet 0/0/0/1.1

  evi 6005
!
```

EVPN Modes

The following EVPN modes are supported:

- Single-homing - Enables you to connect a customer edge (CE) device to one provider edge (PE) device.
- Multihoming - Enables you to connect a customer edge (CE) device to more than one provider edge (PE) device. Multihoming ensures redundant connectivity. The redundant PE device ensures that there is no traffic disruption when there is a network failure. Following are the types of multihoming:
 - Single-Active - In this mode, only a single PE among a group of PEs attached to the particular Ethernet-Segment is allowed to forward traffic to and from that Ethernet Segment.
 - All-Active - In this mode, all PEs attached to the particular Ethernet-Segment is allowed to forward traffic to and from that Ethernet Segment.
 - Port-Active - In this mode, only the PE which is in the active mode sends and receives the traffic. This mode supports single-active redundancy load balancing at the port-level or the interface-level.

EVPN Port-Active Multihoming

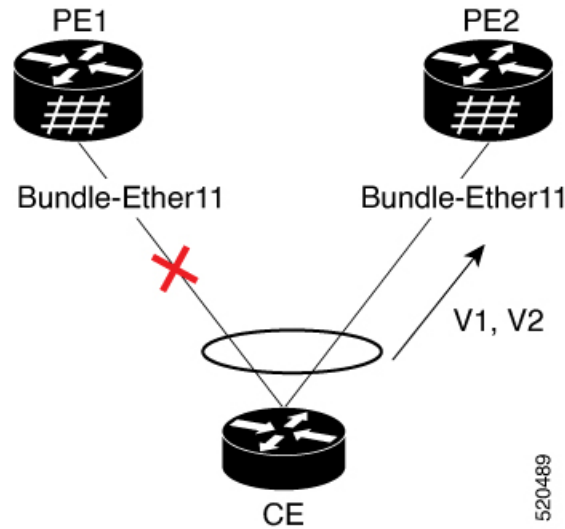
The EVPN Port-Active Multihoming feature supports single-active redundancy load balancing at the port-level or the interface-level. You can use this feature when you want to forward the traffic to a specific interface, rather than have a per-flow load balancing across multiple PE routers. This feature provides a faster convergence during a link failure. This feature enables protocol simplification as only one of the physical ports is active at a given time. You can enable this feature only on bundle interfaces.

EVPN port-active provides protocol simplification compared to Inter-Chassis Communication Protocol (ICCP), which runs on top of Label Distribution Protocol (LDP). You can use this feature as an alternative to multi-chassis link aggregation group (MC-LAG) with ICCP.

Also, you can use this feature when you want certain QoS features to work.

This feature allows one of the PEs to be in active mode and another in the standby mode at the port-level. Only the PE which is in the active mode sends and receives the traffic. The other PE remains in the standby mode. The PEs use the Designated Forwarder (DF) election mechanism to determine which PE must be in the active mode and which must be in the standby mode. You can use either modulo or Highest Random Weight (HRW) algorithm for per port DF election. By default, the modulo algorithm is used for per port DF election.

Figure 2: EVPN Port-Active Multihoming



Consider a topology where the customer edge device (CE) is multihomed to provider edge devices, PE1 and PE2. Use single link aggregation at the CE. Only one of the two interfaces is in the forwarding state, and the other interface is in the standby state. In this topology, PE2 is in the active mode and PE1 is in the standby mode. Hence, PE2 carries traffic from the CE. All services on the PE2 interface operate in the active mode. All services on the PE1 operate in the standby mode.

If the interface is running LACP, then the standby sets the LACP state to Out-of-Service (OOS) instead of bringing the interface state down. This state enables better convergence on standby to active transition.

If you remove the port-active configuration on both PE1 and PE2 and then add back the port-active configuration on both the PEs, PE2 is chosen as an active interface again.

EVPN port-active is compatible with the following services:

- L2 bridging
- L3 gateway
- L2VPN VPLS
- EVPN ELAN
- EVPN IRB
- L2VPN VPWS
- EVPN VPWS
- FXC

This feature supports both L2 and L3 port-active functionality. L2 and L3 port-active can coexist on the same bundle. For example, if you configure port-active on a bundle, the bundle can have a mix of both L3 subinterfaces and L2 subinterfaces participating in the services mentioned above.

Configure EVPN Port-Active Multihoming

Perform this task to configure EVPN port-active multihoming.

Configure the same ESI on both the routers. Configure Ethernet-Segment in port-active load-balancing mode on peering PEs for a specific interface.

Configuration Example

```
/* PE1 and PE2 Configuration */

Router#configure
Router(config)#interface Bundle-Ether11
Router(config-if)#lACP system mac 3637.3637.3637
Router(config-if)#exit

Router(config)#evpn
Router(config-evpn)#interface Bundle-Ether11
Router(config-evpn-ac)#ethernet-segment
Router(config-evpn-ac-es)#identifier type 0 11.11.11.11.11.00.11.11.11
Router(config-evpn-ac-es)#load-balancing-mode port-active
Router(config-evpn-ac-es)#commit

/* If you want enable L3 port-active, configure the IP address */
Router#configure
Router(config)#interface Bundle-Ether11
Router(config-if)#ipv4 address 10.0.0.1 255.0.0.0
Router(config-if)#ipv6 address 10::1/64
Router(config-if)#commit
```

Running Configuration

This section shows port-active running configuration.

```
configure
 interface Bundle-Ether11
   lACP system mac 3637.3637.3637
   !

evpn
 interface Bundle-Ether11
   ethernet-segment
     identifier type 0 11.11.11.11.11.00.11.11.11
     load-balancing-mode port-active
   !
 !
 interface Bundle-Ether11
   ipv4 address 10.0.0.1 255.0.0.0
   ipv6 address 10::1/64
   !
 !
```

Verification

Verify that you have configured the Port-Active Multihoming feature successfully.

```
Router:PE2#show bundle bundle-ether 11

Bundle-Ether11
  Status:                               Up
  Local links <active/standby/configured>: 1 / 0 / 1
  Local bandwidth <effective/available>: 1000000 (1000000) kbps
  MAC address (source):                   02b4.3cb4.a004 (Chassis pool)
```

```

Inter-chassis link:                No
Minimum active links / bandwidth:  1 / 1 kbps
Maximum active links:              64
Wait while timer:                  2000 ms
Load balancing:
  Link order signaling:             Not configured
  Hash type:                        Default
  Locality threshold:              None
LACP:
  Flap suppression timer:          Off
  Cisco extensions:                 Disabled
  Non-revertive:                   Disabled
mLACP:
  IPv4 BFD:                         Not configured
  IPv6 BFD:                         Not configured

```

Port	Device	State	Port ID	B/W, kbps
Gi0/2/0/8	Local	Active	0x8000, 0x0006	1000000

Link is Active

/* PE2 is in the active mode, hence the status shows as Up and the Link as Active. */

Router:PE1#show bundle bundle-ether 11

```

Bundle-Ether11
Status:                LACP OOS (out of service)
Local links <active/standby/configured>:  0 / 1 / 1
Local bandwidth <effective/available>:    0 (0) kbps
MAC address (source):  02cf.94c1.0a04 (Chassis pool)
Inter-chassis link:    No
Minimum active links / bandwidth:        1 / 1 kbps
Maximum active links:  64
Wait while timer:      2000 ms
Load balancing:
  Link order signaling: Not configured
  Hash type:            Default
  Locality threshold:  None
LACP:
  Flap suppression timer: Off
  Cisco extensions:       Disabled
  Non-revertive:         Disabled
mLACP:
  IPv4 BFD:              Not configured
  IPv6 BFD:              Not configured

```

Port	Device	State	Port ID	B/W, kbps
Gi0/2/0/7	Local	Standby	0x8000, 0x0006	1000000

Link is in standby due to bundle out of service state

/* PE1 is in the standby mode, hence the status shows as LACP OOS (out of service) and the Link is in standby due to bundle out of service state. */

Router:CE#show bundle bundle-ether 11

```

Bundle-Ether11
Status:                Up
Local links <active/standby/configured>:  1 / 0 / 2
Local bandwidth <effective/available>:    1000000 (1000000) kbps
MAC address (source):  02ff.566c.be04 (Chassis pool)
Inter-chassis link:    No
Minimum active links / bandwidth:        1 / 1 kbps
Maximum active links:  64

```

```

Wait while timer:                2000 ms
Load balancing:
  Link order signaling:           Not configured
  Hash type:                      Default
  Locality threshold:             None
LACP:                             Operational
  Flap suppression timer:         Off
  Cisco extensions:                Disabled
  Non-revertive:                  Disabled
mLACP:                             Not configured
IPv4 BFD:                         Not configured
IPv6 BFD:                         Not configured

```

Port	Device	State	Port ID	B/W, kbps
Gi0/0/0/8	Local	Active	0x8000, 0x0006	1000000
Link is Active				
Gi0/0/0/16	Local	Negotiating	0x8000, 0x000b	1000000
Partner is not Synchronized (Waiting, Standby, or LAG ID mismatch)				

```
Router:PE2#show evpn ethernet-segment interface BE11 detail
```

```
/* The following output shows that the port-active mode is configured and the port is in
the UP state. */
```

```

Ethernet Segment Id           Interface           Nexthops
-----
0011.1111.1111.0011.1111 BE11
                                192.168.0.2
                                192.168.0.3

ES to BGP Gates      : Ready
ES to L2FIB Gates   : Ready
Main port            :
  Interface name     : Bundle-Ether11
  Interface MAC      : 02b4.3cb4.a004
  IfHandle           : 0x00004170
  State              : Up
  Redundancy         : Not Defined
ESI type             : 0
  Value              : 11.1111.1111.0011.1111
ES Import RT        : 1111.1111.1100 (from ESI)
Source MAC          : 0000.0000.0000 (N/A)
Topology            :
  Operational        : MH
  Configured         : Port-Active
Service Carving     : Auto-selection
  Multicast          : Disabled
Convergence         :
  Mobility-Flush     : Count 0, Skip 0, Last n/a
Peering Details     : 2 Nexthops
  192.168.0.2 [MOD:P:7fff]
  192.168.0.3 [MOD:P:00]
Service Carving Results:
  Forwarders        : 0
  Elected           : 0
  Not Elected       : 0
EVPN-VPWS Service Carving Results:
  Primary           : 0
  Backup            : 0
  Non-DF            : 0
MAC Flushing mode   : STP-TCN
Peering timer       : 3 sec [not running]
Recovery timer      : 20 sec [not running]
Carving timer       : 0 sec [not running]
Local SHG label     : None

```

```
Remote SHG labels : 0
Access signal mode: Bundle OOS (Default)
```

```
Router:PE1#show evpn ethernet-segment interface BE11 detail
```

```
/* The following output shows that the por-active mode is configured and the port is in the
Standby state. */
```

```

Ethernet Segment Id          Interface          Nexthops
-----
0011.1111.1111.0011.1111 BE11
                                192.168.0.2
                                192.168.0.3

ES to BGP Gates      : Ready
ES to L2FIB Gates   : Ready
Main port           :
  Interface name    : Bundle-Ether11
  Interface MAC     : 02cf.941c.0a04
  IfHandle          : 0x00004170
  State             : Standby
  Redundancy        : Not Defined
ESI type            : 0
  Value             : 11.1111.1111.0011.1111
ES Import RT        : 1111.1111.1100 (from ESI)
Source MAC          : 0000.0000.0000 (N/A)
Topology            :
  Operational       : MH
  Configured        : Port-Active
Service Carving     : Auto-selection
  Multicast         : Disabled
Convergence          :
  Mobility-Flush    : Count 0, Skip 0, Last n/a
Peering Details     : 2 Nexthops
  192.168.0.2 [MOD:P:00]
  192.168.0.3 [MOD:P:7fff]
Service Carving Results:
  Forwarders       : 0
  Elected          : 0
  Not Elected      : 0
EVPN-VPWS Service Carving Results:
  Primary           : 0
  Backup            : 0
  Non-DF            : 0
MAC Flushing mode   : STP-TCN
Peering timer       : 3 sec [not running]
Recovery timer      : 20 sec [not running]
Carving timer       : 0 sec [not running]
Local SHG label     : None
Remote SHG labels   : 0
Access signal mode  : Bundle OOS (Default)

```

EVPN MPLS Seamless Integration with VPLS

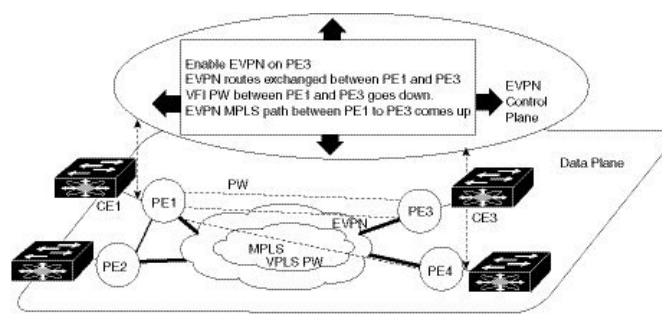
Migrate VPLS Network to EVPN Network through Seamless Integration

In EVPN network, VPN instances are identified by EVPN instance ID (EVI-ID). Similar to other L2VPN technologies, EVPN instances are also associated with route-targets and route-distinguisher. EVPN uses control plane for learning and propagating MAC unlike traditional VPLS, where MAC is learnt in the data

plane (learns using "flood and learn technique"). In EVPN, MAC routes are carried by MP-BGP protocol. In EVPN enabled PEs, PEs import the MAC route along with the label to their respective EVPN forwarding table only if their route targets (RTs) match. An EVPN PE router is capable of performing VPLS and EVPN L2 bridging in the same VPN instance. When both EVPN and BGP-AD PW are configured in a VPN instance, the EVPN PEs advertise the BGP VPLS auto-discovery (AD) route as well as the BGP EVPN Inclusive Multicast route (type-3) for a given VPN Instance. Route type-3 referred to as ingress replication multicast route, is used to send broadcast, unknown unicast, and multicast (BUM) traffic. Other remote PEs import type-3 routes for the same VPN instance only if the sending PE RTs match with their configured RT. Thus, at the end of these route-exchanges, EVPN capable PEs discover all other PEs in the VPN instance and their associated capabilities. The type-3 routes used by PE to send its BUM traffic to other PEs ensure that PEs with the same RTs receive the BUM traffic. EVPN advertises the customer MAC address using type-2 route.

EVPN MPLS Seamless Integration with VPLS allows you to upgrade the VPLS PE routers to EVPN one by one without any network service disruption. Consider the following topology where PE1, PE2, PE3, and PE4 are interconnected in a full-meshed network using VPLS PW.

Figure 3: EVPN MPLS Seamless Integration with VPLS



The EVPN service can be introduced in the network one PE node at a time. The VPLS to EVPN migration starts on PE1 by enabling EVPN in a VPN instance of VPLS service. As soon as EVPN is enabled, PE1 starts advertising EVPN inclusive multicast route to other PE nodes. Since PE1 does not receive any inclusive multicast routes from other PE nodes, VPLS pseudo wires between PE1 and other PE nodes remain active. PE1 keeps forwarding traffic using VPLS pseudo wires. At the same time, PE1 advertises all MAC address learned from CE1 using EVPN route type-2. In the second step, EVPN is enabled in PE3. PE3 starts advertising inclusive multicast route to other PE nodes. Both PE1 and PE3 discover each other through EVPN routes. As a result, PE1 and PE3 shut down the pseudo wires between them. EVPN service replaces VPLS service between PE1 and PE3. At this stage, PE1 keeps running VPLS service with PE2 and PE4. It starts EVPN service with PE3 in the same VPN instance. This is called EVPN seamless integration with VPLS. The VPLS to EVPN migration then continues to remaining PE nodes. In the end, all four PE nodes are enabled with EVPN service. VPLS service is completely replaced with EVPN service in the network. All VPLS pseudo wires are shut down.

Configure EVPN on the Existing VPLS Network

Perform the following tasks to configure EVPN on the existing VPLS network.

- Configure L2VPN EVPN address-family
- Configure EVI and corresponding BGP route-targets under EVPN configuration mode
- Configure EVI under a bridge-domain

See [EVI Configuration Under L2VPN Bridge-Domain, on page 19](#) section for how to migrate various VPLS-based network to EVPN.

Configure L2 EVPN Address-Family

Perform this task to enable EVPN address family under both BGP and participating neighbor.

Configuration Example

```
Router# configure
Router(config)#router bgp 65530
Router(config-bgp)#nsr
Router(config-bgp)#bgp graceful-restart
Router(config-bgp)#bgp router-id 200.0.1.1
Router(config-bgp)#address-family l2vpn evpn
Router(config-bgp-af)#exit
Router(config-bgp)#neighbor 200.0.4.1
Router(config-bgp-nbr)#remote-as 65530
Router(config-bgp-nbr)#update-source Loopback0
Router(config-bgp-nbr)#address-family l2vpn evpn
Router(config-bgp-nbr-af)#commit
```

Running Configuration

```
configure
router bgp 65530
  nsr
  bgp graceful-restart
  bgp router-id 200.0.1.1
  address-family l2vpn evpn
  !
  neighbor 200.0.4.1
    remote-as 65530
    update-source Loopback0
    address-family l2vpn evpn
  !
!
```

Configure EVI and Corresponding BGP Route Target under EVPN Configuration Mode

Perform this task to configure EVI and define the corresponding BGP route targets. Also, configure advertise-mac, else the MAC routes (type-2) are not advertised.

Configuration Example

```
Router# configure
Router(config)#evpn
Router(config-evpn)#evi 1
Router(config-evpn-evi-bgp)#bgp
Router(config-evpn-evi-bgp)#table-policy spp-basic-6
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)#exit
```

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

Running Configuration

```
configure
 evpn
  evi
    bgp
      table-policy spp-basic-6
      route-target import 100:6005
      route-target export 100:6005
    !
    advertise-mac
  !
!
```

Configure EVI under a Bridge Domain

Perform this task to configure EVI under the corresponding L2VPN bridge domain.

Configuration 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 GigabitEthernet0/2/0/0.1
Router(config-l2vpn-bg-bd-ac)#exit
Router(config-l2vpn-bg-bd)#evi 1
Router(config-l2vpn-bg-bd-evi)#exit
Router(config-l2vpn-bg-bd)#vfi v1
Router(config-l2vpn-bg-bd-vfi)#neighbor 10.1.1.2 pw-id 1000
Router(config-l2vpn-bg-bd-vfi-pw)#mpls static label local 20001 remote 10001
Router(config-l2vpn-bg-bd-vfi-pw)#commit
```

Running Configuration

```
configure
 l2vpn
  bridge group bg1
  bridge-domain bd1
  interface GigabitEthernet0/2/0/0.1
  !
  evi 1
  !
  vfi v1
  neighbor 10.1.1.2 pw-id 1000
  mpls static label local 20001 remote 10001
  !
  !
  evi 1
!
```

EVI Configuration Under L2VPN Bridge-Domain

The following examples show EVI configuration under L2VPN bridge-domain for various VPLS-based networks:

MPLS Static Labels Based VPLS

```
l2vpn
bridge group bg1
bridge-domain bd-1-1
  interface GigabitEthernet0/2/0/0.1
  !
  vfi vfi-1-1
  neighbor 200.0.2.1 pw-id 1200001
  mpls static label local 20001 remote 10001
  !
  neighbor 200.0.3.1 pw-id 1300001
  mpls static label local 30001 remote 10001
  !
  neighbor 200.0.4.1 pw-id 1400001
  mpls static label local 40001 remote 10001
  !
  !
  evi 1
  !
```

AutoDiscovery BGP and BGP Signalling Based VPLS

```
l2vpn
bridge group bg1
bridge-domain bd-1-2
  interface GigabitEthernet0/2/0/0.2
  !
  vfi vfi-1-2
  vpn-id 2
  autodiscovery bgp
  rd 101:2
  route-target 65530:200
  signaling-protocol bgp
  ve-id 11
  ve-range 16
  !
  !
  evi 2
  !
```

Targeted LDP-Based VPLS

```
bridge-domain bd-1-4
  interface GigabitEthernet0/2/0/0.4
  !
  vfi vfi-1-4
  neighbor 200.0.2.1 pw-id 1200004
  !
  neighbor 200.0.3.1 pw-id 1300004
  !
  neighbor 200.0.4.1 pw-id 1400004
  !
```

```
evi 3
!
```

Verify EVPN Configuration

Use the following commands to verify EVPN configuration and MAC advertisement. Verify EVPN status, AC status, and VFI status.

- show l2vpn bridge-domain
- show evpn summary
- show bgp rt l2vpn evpn
- show evpn evi
- show l2route evpn mac all

```
Router#show l2vpn bridge-domain bd-name bd-1-1
Mon Feb 20 21:03:40.244 EST
Legend: pp = Partially Programmed.
Bridge group: bg1, bridge-domain: bd-1-1, id: 0, state: up, ShgId: 0, MSTi: 0
Aging: 300 s, MAC limit: 4000, Action: none, Notification: syslog
Filter MAC addresses: 0
ACs: 1 (1 up), VFIs: 1, PWs: 3 (2 up), PBBs: 0 (0 up), VNIs: 0 (0 up)
List of EVPNs:
  EVPN, state: up
List of ACs:
  Gi0/2/0/0.1, state: up, Static MAC addresses: 0, MSTi: 2
List of Access PWs:
List of VFIs:
  VFI vfi-1-1 (up)
    Neighbor 200.0.2.1 pw-id 1200001, state: up, Static MAC addresses: 0
    Neighbor 200.0.3.1 pw-id 1300001, state: down, Static MAC addresses: 0
    Neighbor 200.0.4.1 pw-id 1400001, state: up, Static MAC addresses: 0
  List of Access VFIs:
  When PEs are evpn enabled, pseudowires that are associated with that BD will be brought
  down. The VPLS BD pseudowires are always up.
```

Verify the number of EVI's configured, local and remote MAC-routes that are advertised.

```
Router#show evpn summary
Mon Feb 20 21:05:16.755 EST
-----
Global Information
-----
Number of EVIs                : 6
Number of Local EAD Entries    : 0
Number of Remote EAD Entries   : 0
Number of Local MAC Routes     : 4
    MAC                        : 4
    MAC-IPv4                    : 0
    MAC-IPv6                    : 0
Number of Local ES:Global MAC  : 1
Number of Remote MAC Routes    : 0
    MAC                        : 0
    MAC-IPv4                    : 0
    MAC-IPv6                    : 0
Number of Remote SOO MAC Routes : 0
Number of Local IMCAST Routes  : 4
```

```

Number of Remote IMCAST Routes      : 4
Number of Internal Labels            : 0
Number of ES Entries                 : 1
Number of Neighbor Entries           : 4
EVPN Router ID                       : 200.0.1.1
BGP ASN                              : 65530
PBB BSA MAC address                  : 0026.982b.c1e5
Global peering timer                 :      3 seconds
Global recovery timer                 :     30 seconds

```

Verify EVPN route-targets.

```

Router#show bgp rt l2vpn evpn
Mon Feb 20 21:06:18.882 EST
EXTCOMM      IMP/EXP
RT:65530:1   1 / 1
RT:65530:2   1 / 1
RT:65530:3   1 / 1
RT:65530:4   1 / 1
Processed 4 entries

```

Locally learnt MAC routes can be viewed by forwarding table
show l2vpn forwarding bridge-domain mac-address location 0/0/cpu0
To Resynchronize MAC table from the Network Processors, use the command...
l2vpn resynchronize forwarding mac-address-table location <r/s/i>

Mac Address	Type	Learned from/Filtered on	LC learned	Resync Age/Last Change	Mapped to
0033.0000.0001	dynamic	Gi0/2/0/0.1	N/A	20 Feb 21:06:59	N/A
0033.0000.0002	dynamic	Gi0/2/0/0.2	N/A	20 Feb 21:06:59	N/A
0033.0000.0003	dynamic	Gi0/2/0/0.3	N/A	20 Feb 21:04:29	N/A
0033.0000.0004	dynamic	Gi0/2/0/0.4	N/A	20 Feb 21:06:59	N/A

The remote routes learned via evpn enabled BD
show l2vpn forwarding bridge-domain mac-address location 0/0/\$
To Resynchronize MAC table from the Network Processors, use the command...
l2vpn resynchronize forwarding mac-address-table location <r/s/i>

Mac Address	Type	Learned from/Filtered on	LC learned	Resync Age/Last Change	Mapped to
0033.0000.0001	EVPN	BD id: 0	N/A	N/A	N/A
0033.0000.0002	EVPN	BD id: 1	N/A	N/A	N/A
0033.0000.0003	EVPN	BD id: 2	N/A	N/A	N/A
0033.0000.0004	EVPN	BD id: 3	N/A	N/A	N/A

Verify EVPN MAC routes pertaining to specific VPN instance.

```

Router#show evpn evi vpn-id 1 mac
Mon Feb 20 21:36:23.574 EST

EVI      MAC address      IP address      Nexthop
Label
-----

```

```
1      0033.0000.0001      ::      200.0.1.1      45106
```

Verify L2 routing.

```
Router#show l2route evpn mac all
Mon Feb 20 21:39:43.953 EST
Topo ID  Mac Address      Prod   Next Hop(s)
-----  -
0         0033.0000.0001 L2VPN 200.0.1.1/45106/ME
1         0033.0000.0002 L2VPN 200.0.1.1/45108/ME
2         0033.0000.0003 L2VPN 200.0.1.1/45110/ME
3         0033.0000.0004 L2VPN 200.0.1.1/45112/ME
```

Verify EVPN route-type 2 routes.

```
Router#show bgp l2vpn evpn route-type 2
Mon Feb 20 21:43:23.616 EST
BGP router identifier 200.0.3.1, local AS number 65530
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0   RD version: 0
BGP main routing table version 21
BGP NSR Initial initsync version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
                i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop          Metric LocPrf Weight Path
Route Distinguisher: 200.0.1.1:1
*>i[2][0][48][0033.0000.0001][0]/104
                200.0.1.1                      100      0 i
Route Distinguisher: 200.0.1.1:2
*>i[2][0][48][0033.0000.0002][0]/104
                200.0.1.1                      100      0 i
Route Distinguisher: 200.0.1.1:3
*>i[2][0][48][0033.0000.0003][0]/104
                200.0.1.1                      100      0 i
Route Distinguisher: 200.0.1.1:4
*>i[2][0][48][0033.0000.0004][0]/104
                200.0.1.1                      100      0 i
Route Distinguisher: 200.0.3.1:1 (default for vrf bd-1-1)
*>i[2][0][48][0033.0000.0001][0]/104
                200.0.1.1                      100      0 i
Route Distinguisher: 200.0.3.1:2 (default for vrf bd-1-2)
*>i[2][0][48][0033.0000.0002][0]/104
                200.0.1.1                      100      0 i
Route Distinguisher: 200.0.3.1:3 (default for vrf bd-1-3)
*>i[2][0][48][0033.0000.0003][0]/104
                200.0.1.1                      100      0 i
Route Distinguisher: 200.0.3.1:4 (default for vrf bd-1-4)
*>i[2][0][48][0033.0000.0004][0]/104
                200.0.1.1                      100      0 i

Processed 8 prefixes, 8 paths
```

Verify inclusive multicast routes and route-type 3 routes.

```
Router#show bgp l2vpn evpn route-type 3
```

```

Mon Feb 20 21:43:33.970 EST
BGP router identifier 200.0.3.1, local AS number 65530
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0   RD version: 0
BGP main routing table version 21
BGP NSR Initial initsync version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
               i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop          Metric LocPrf Weight Path
Route Distinguisher: 200.0.1.1:1
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
Route Distinguisher: 200.0.1.1:2
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
Route Distinguisher: 200.0.1.1:3
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
Route Distinguisher: 200.0.1.1:4
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
Route Distinguisher: 200.0.3.1:1 (default for vrf bd-1-1)
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
*> [3][0][32][200.0.3.1]/80
                0.0.0.0                    0 i
Route Distinguisher: 200.0.3.1:2 (default for vrf bd-1-2)
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
*> [3][0][32][200.0.3.1]/80
                0.0.0.0                    0 i
Route Distinguisher: 200.0.3.1:3 (default for vrf bd-1-3)
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
*> [3][0][32][200.0.3.1]/80
                0.0.0.0                    0 i
Route Distinguisher: 200.0.3.1:4 (default for vrf bd-1-4)
*>i[3][0][32][200.0.1.1]/80
                200.0.1.1                100      0 i
*> [3][0][32][200.0.3.1]/80
                0.0.0.0                    0 i

```

Clear Forwarding Table

To clear an L2VPN forwarding table at a specified location, you can use the **clear l2vpn forwarding table** command. When BVI is present in the bridge domain, you might experience traffic loss during the command execution. Refer the following work-around to resolve such issues.

When you encounter such issues, delete the BVI and roll back the action. As a result, the traffic on the BVI returns to normal state. The following example shows how to delete the BVI and perform roll back action:

```

Router#clear l2vpn forwarding table location 0/0/CPU0
Fri Mar 24 09:34:02.083 UTC
Router(config)#no int BVI100
Router(config)#commit
Router#roll configuration las 1

```

```
Wed Dec 16 18:26:52.869 UTC
Loading Rollback Changes.
Loaded Rollback Changes in 1 sec
Committing
```



Note We can also clear the forwarding table by shutting and unshutting the interface.

EVPN Features

This section lists the supported EVPN features and how to configure them:

Configure EVPN MAC Address Limit

To configure EVPN MAC address limit, the following restrictions are applicable:

- Remote MAC addresses are programmed in the hardware irrespective of whether the MAC address limit is configured or not.
- MAC address limit can be modified correctly only when the device is not actively learning any MAC addresses. This is an expected behavior.
- When the MAC learning is enabled, you can configure the MAC address limit up to a maximum of six. However, when the MAC learning is disabled, you can configure the MAC address limit up to a maximum of five.
- The **clear l2vpn mac address table** command is not supported. The MAC address table is cleared when **shut** or **no shutdown** is performed on an attachment circuit interface or sub interface, or when the MAC aging timer expires.
- You can configure both MAC limit Action and MAC notification. However, the configuration does not take into effect as the functionality is not supported.

Configuration Example

Perform this task to configure EVPN MAC address limit.

This table lists the MAC address limit parameters and values that are configured:

Parameter	Value
MAC address limit	50
MAC limit threshold	80%

```
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group EVPN-BG-SH
Router(config-l2vpn-bg)# bridge-domain EVPN_2701
Router(config-l2vpn-bg-bd)# mac
Router(config-l2vpn-bg-bd-mac)# limit
Router(config-l2vpn-bg-bd-mac-limit)# maximum 50
```



```

Router(config-l2vpn-bg-bd-mac-limit)# exit
Router(config-l2vpn-bg-bd)# exit
Router(config-l2vpn-bg)# exit
Router(config-l2vpn)# mac limit threshold 80
Router(config-l2vpn)# commit

```

Running Configuration

```

l2vpn
bridge group EVPN-BG-SH
  bridge-domain EVPN_2701
    mac
      limit
        maximum 50
    !
  !
!
mac limit threshold 80
commit

```

Verification

Verify the EVPN MAC address limit parameters are set as described in above table:

```

Router# show l2vpn bridge-domain bd-name EVPN_2701 detail
Legend: pp = Partially Programmed.
Bridge group: EVPN-BG-SH, bridge-domain: EVPN_2701, id: 25, 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: 50, Action: none, Notification: syslog
MAC limit reached: no, threshold: 80%
  MAC port down flush: enabled
  MAC Secure: disabled, Logging: disabled
  Split Horizon Group: none
  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: 26
  Filter MAC addresses:
  P2MP PW: disabled
  Create time: 21/04/2019 16:28:05 (2d23h ago)
  No status change since creation
  ACs: 1 (1 up), VFIs: 0, PWs: 0 (0 up), PBBs: 0 (0 up), VNIs: 0 (0 up)
  List of EVPNs:
    EVPN, state: up
      evi: 6101

```

```

XC ID 0x8000040c
Statistics:
  packets: received 0 (unicast 0), sent 0
  bytes: received 0 (unicast 0), sent 0
  MAC move: 0
List of ACs:
AC: Bundle-Ether101.2701, state is up, active in RG-ID 101
  Type VLAN; Num Ranges: 1
  Rewrite Tags: [1000, 2000]
  VLAN ranges: [2701, 2701]
  MTU 9112; XC ID 0xa000060b; interworking none; MSTi 6
  MAC learning: enabled
  Flooding:
    Broadcast & Multicast: enabled
    Unknown unicast: enabled
  MAC aging time: 300 s, Type: inactivity
MAC limit: 50, Action: none, Notification: syslog
MAC limit reached: no, threshold: 80%
  MAC port down flush: enabled
  MAC Secure: disabled, Logging: disabled
  Split Horizon Group: none
  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:
    Broadcast: enabled(160000 pps)
    Multicast: enabled(160000 pps)
    Unknown unicast: enabled(160000 pps)
  Static MAC addresses:
  Statistics:
    packets: received 0 (multicast 0, broadcast 0, unknown unicast 0, unicast 0), sent
0
    bytes: received 0 (multicast 0, broadcast 0, unknown unicast 0, unicast 0), sent 0
    MAC move: 0
  Storm control drop counters:
    packets: broadcast 0, multicast 0, unknown unicast 0
    bytes: broadcast 0, multicast 0, unknown unicast 0
  Dynamic ARP inspection drop counters:
    packets: 0, bytes: 0
  IP source guard drop counters:
    packets: 0, bytes: 0
List of Access PWs:
List of VFIs:
List of Access VFIs:

```

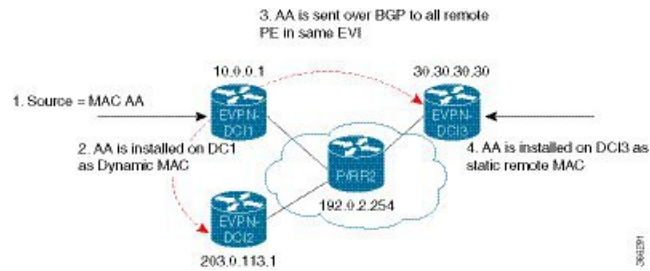
EVPN Software MAC Learning

The MAC addresses learned on one device needs to be learned or distributed on the other devices in a VLAN. EVPN Software MAC Learning feature enables the distribution of the MAC addresses learned on one device to the other devices connected to a network. The MAC addresses are learnt from the remote devices using BGP.



Note A device can contain up to 128K MAC address entries. A bridge domain on a device can contain up to 64K MAC address entries.

Figure 4: EVPN Software MAC Learning



The above figure illustrates the process of software MAC learning. The following are the steps involved in the process:

1. Traffic comes in on one port in the bridge domain.
2. The source MAC address (AA) is learnt on the PE and is stored as a dynamic MAC entry.
3. The MAC address (AA) is converted into a type-2 BGP route and is sent over BGP to all the remote PEs in the same EVI.
4. The MAC address (AA) is updated on the PE as a remote MAC address.

Configure EVPN Software MAC Learning

The following section describes how you can configure EVPN Software MAC Learning:



Note On EVPN bridge domain, the Cisco NCS 5500 router does not support control word and does not enable control word by default.



Note The router does not support flow-aware transport (FAT) pseudowire.

```

/* Configure bridge domain. */

RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group EVPN_SH
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain EVPN_2001
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface TenGigE0/4/0/10.2001
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface BundleEther 20.2001
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# storm-control broadcast pps 10000 ← Enabling
storm-control is optional
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# exit
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# evi 2001
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-evi)# commit

/* Configure address family session in BGP. */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# router bgp 200
RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 209.165.200.227
RP/0/RSP0/CPU0:router(config-bgp)# address-family l2vpn evpn

```

```
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.10.10.10
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 200
RP/0/RSP0/CPU0:router(config-bgp-nbr)# description MPLSFACINGPEER
RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source Loopback 0
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family l2vpn evpn
```

Supported Modes for EVPN Software MAC Learning

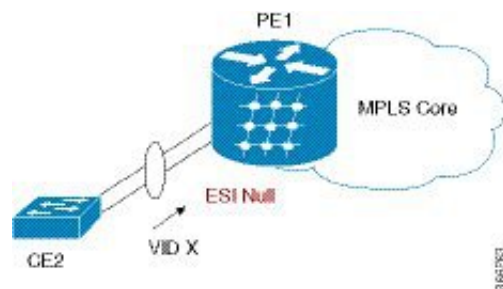
The following are the modes in which EVPN Software MAC Learning is supported:

- Single Home Device (SHD) or Single Home Network (SHN)
- Dual Home Device (DHD)—All Active Load Balancing

Single Home Device or Single Home Network Mode

The following section describes how you can configure EVPN Software MAC Learning feature in single home device or single home network (SHD/SHN) mode:

Figure 5: Single Home Device or Single Home Network Mode



In the above figure, the PE (PE1) is attached to Ethernet Segment using bundle or physical interfaces. Null Ethernet Segment Identifier (ESI) is used for SHD/SHN.

Configure EVPN in Single Home Device or Single Home Network Mode

This section describes how you can configure EVPN Software MAC Learning feature in single home device or single home network mode.

```
/* Configure bridge domain. */

RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group EVPN_ALL_ACTIVE
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain EVPN_2001
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface Bundle-Ether1.2001
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# evi 2001

/* Configure advertisement of MAC routes. */

RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# evi 2001
RP/0/RSP0/CPU0:router(config-evpn-evi)# advertise-mac

/* Configure address family session in BGP. */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router#(config)# router bgp 200
RP/0/RSP0/CPU0:router#(config-bgp)# bgp router-id 09.165.200.227
```

```
RP/0/RSP0/CPU0:router# (config-bgp) # address-family l2vpn evpn
RP/0/RSP0/CPU0:router# (config-bgp) # neighbor 10.10.10.10
RP/0/RSP0/CPU0:router# (config-bgp-nbr) # remote-as 200
RP/0/RSP0/CPU0:router# (config-bgp-nbr) # description MPLSFACING-PEER
RP/0/RSP0/CPU0:router# (config-bgp-nbr) # update-source Loopback 0
RP/0/RSP0/CPU0:router# (config-bgp-nbr) # address-family l2vpn evpn
```

Running Configuration

```
l2vpn
bridge group EVPN_ALL_ACTIVE
  bridge-domain EVPN_2001
  interface BundleEther1.2001
  evi 2001
!
evpn
  evi 2001
  advertise-mac
!
router bgp 200 bgp
  router-id 40.40.40.40
  address-family l2vpn evpn
  neighbor 10.10.10.10
  remote-as 200 description MPLS-FACING-PEER
  updatesource Loopback0
  addressfamily l2vpn evpn
```

Verification

Verify EVPN in single home devices.

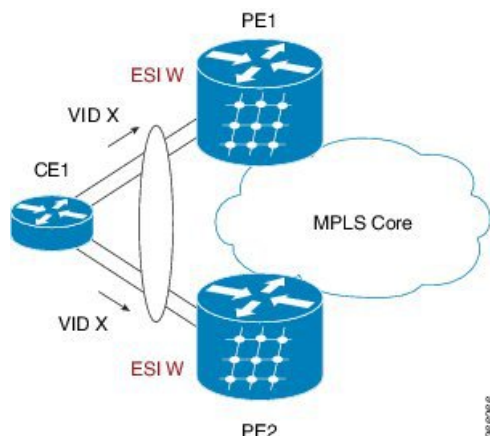
```
RP/0/RSP0/CPU0:router# show evpn ethernet-segment interface Te0/4/0/10 detail

Ethernet Segment Id      Interface      Nexthops
-----
N/A                      Te0/4/0/10   20.20.20.20
.....
Topology :
Operational : SH
Configured : Single-active (AApS) (default)
```

Dual Home Device—All-Active Load Balancing Mode

The following section describes how you can configure EVPN Software MAC Learning feature in dual home device (DHD) in all-active load balancing mode:

Figure 6: Dual Home Device —All-Active Load Balancing Mode



All-active load-balancing is known as Active/Active per Flow (AApF). In the above figure, identical Ethernet Segment Identifier is used on both EVPN PEs. PEs are attached to Ethernet Segment using bundle interfaces. In the CE, single bundles are configured towards two EVPN PEs. In this mode, the MAC address that is learnt is stored on both PE1 and PE2. Both PE1 and PE2 can forward the traffic within the same EVI.

Configure EVPN Software MAC Learning in Dual Home Device—All-Active Mode

This section describes how you can configure EVPN Software MAC Learning feature in dual home device—all-active mode:

```

/* Configure bridge domain. */

RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group EVPN_ALL_ACTIVE
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain EVPN_2001
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface Bundle-Ether1
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# evi 2001

/* Configure advertisement of MAC routes. */

RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# evi 2001
RP/0/RSP0/CPU0:router(config-evpn-evi)# advertise-mac
RP/0/RSP0/CPU0:router(config-evpn-evi)# exit
RP/0/RSP0/CPU0:router(config-evpn)# interface Bundle-Ether1
RP/0/RSP0/CPU0:router(config-evpn-ac)# ethernet-segment
RP/0/RSP0/CPU0:router(config-evpn-ac-es)# identifier type 0 01.11.00.00.00.00.00.01

/* Configure address family session in BGP. */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router#(config)# router bgp 200
RP/0/RSP0/CPU0:router#(config-bgp)# bgp router-id 209.165.200.227
RP/0/RSP0/CPU0:router#(config-bgp)# address-family l2vpn evpn
RP/0/RSP0/CPU0:router#(config-bgp)# neighbor 10.10.10.10
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# remote-as 200
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# description MPLS-FACING-PEER
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# update-source Loopback 0
RP/0/RSP0/CPU0:router#(config-bgp-nbr)# address-family l2vpn evpn

```

```

/* Configure Link Aggregation Control Protocol (LACP) bundle. */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether1
RP/0/RSP0/CPU0:router(config-if)# lacp switchover suppress-flaps 300
RP/0/RSP0/CPU0:router(config-if)# exit

/* Configure VLAN Header Rewrite.*/

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether1 l2transport
RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1q 10
RP/0/RSP0/CPU0:router(config-if)# rewrite ingress tag pop 1 symmetric

```



Note Configure the same mlacp system priority <id> for both the dual homed PE routers to enable all-active load balancing.

Running Configuration

```

l2vpn
bridge group EVPN_ALL_ACTIVE
  bridge-domain EVPN_2001
  interface Bundle-Ether1
  !
  evi 2001
  !
  !
  evpn
  evi 2001
  !
  advertise-mac
  !
  interface Bundle-Ether1
  ethernet-segment
  identifier type 0 01.11.00.00.00.00.00.01
  !
  !
  router bgp 200
  bgp router-id 209.165.200.227
  address-family l2vpn evpn
  !
  neighbor 10.10.10.10
  remote-as 200
  description MPLS-FACING-PEER
  update-source Loopback0
  address-family l2vpn evpn
  !
  interface Bundle-Ether1
  lacp switchover suppress-flaps 300
  load-interval 30
  !
  interface Bundle-Ether1 l2transport
  encapsulation dot1q 2001
  rewrite ingress tag pop 1 symmetric
  !

```

Verification

Verify EVPN in dual home devices in All-Active mode.



Note With the EVPN IRB, the supported label mode is per-VRF.

```
RP/0/RSP0/CPU0:router# show evpn ethernet-segment interface Bundle-Ether 1 carvin$

Ethernet Segment Id      Interface  Nexthops
-----
0100.211b.fce5.df00.0b00  BE1       10.10.10.10
209.165.201.1
Topology :
Operational : MHN
Configured : All-active (AApF) (default)
Primary Services : Auto-selection
Secondary Services: Auto-selection
Service Carving Results:
Forwarders : 4003
Elected : 2002
EVI E : 2000, 2002, 36002, 36004, 36006, 36008
.....
Not Elected : 2001
EVI NE : 2001, 36001, 36003, 36005, 36007, 36009

MAC Flushing mode : Invalid

Peering timer : 3 sec [not running]
Recovery timer : 30 sec [not running]
Local SHG label : 34251
Remote SHG labels : 1
38216 : nexthop 209.165.201.1
```

Verify EVPN Software MAC Learning

Verify the packet drop statistics.



Note Disable CW configuration if any in EVPN peer nodes, as CW is not supported in EVPN Bridging.

```
RP/0/RSP0/CPU0:router# show l2vpn bridge-domain bd-name EVPN_2001 details

Bridge group: EVPN_ALL_ACTIVE, bridge-domain: EVPN_2001, id: 1110,
state: up, ShgId: 0, MSTi: 0
List of EVPNs:
EVPN, state: up
evi: 2001
XC ID 0x80000458
Statistics:
packets: received 28907734874 (unicast 9697466652), sent
76882059953
bytes: received 5550285095808 (unicast 1861913597184), sent
14799781851396
MAC move: 0
List of ACs:
AC: TenGigE0/4/0/10.2001, state is up
Type VLAN; Num Ranges: 1
...
Statistics:
```



```

packets: received 0 (multicast 0, broadcast 0, unknown
unicast 0, unicast 0), sent 45573594908
bytes: received 0 (multicast 0, broadcast 0, unknown unicast
0, unicast 0), sent 8750130222336
MAC move: 0
.....

```

Verify the EVPN EVI information with the VPN-ID and MAC address filter.

```
RP/0/RSP0/CPU0:router# show evpn evi vpn-id 2001 neighbor
```

```

Neighbor IP      vpn-id
-----
209.165.200.225  2001
209.165.201.30   2001

```

Verify the BGP L2VPN EVPN summary.

```
RP/0/RSP0/CPU0:router# show bgp l2vpn evpn summary
```

```

...
Neighbor          Spk   AS      MsgRcvd  MsgSent  TblVer   InQ   OutQ   Up/Down   St/PfxRcd
209.165.200.225  0     200     216739   229871   200781341  0     0     3d00h    348032
209.165.201.30   0     200     6462962  4208831  200781341  10    0     2d22h    35750

```

Verify the MAC updates to the L2FIB table in a line card.

```
RP/0/RSP0/CPU0:router# show l2vpn mac mac all location 0/6/CPU0
```

```

Topo ID          Producer      Next Hop(s)      Mac Address      IP Address
-----
1112            0/6/CPU0      Te0/6/0/1.36001 00a3.0001.0001

```

Verify the MAC updates to the L2FIB table in a route switch processor (RSP).

```
RP/0/RSP0/CPU0:router# show l2vpn mac mac all location 0/6/CPU0
```

```

Topo ID          Producer      Next Hop(s)      Mac Address      IP Address
-----
1112            0/6/CPU0      Te0/6/0/1.36001 00a3.0001.0001

```

Verify the summary information for the MAC address.

```
RP/0/RP0/CPU0:router# show l2vpn forwarding bridge-domain EVPN_ALL_ACTIVE:EVPN_2001
mac-address location 0/6/CPU0
```

```

Mac Address      Type      Learned from/Filtered on      LC learned Resync Age/Last Change
Mapped to
-----
00a3.0001.0001  dynamic  Te0/6/0/1.36001              N/A          01 Sep 10:09:17
N/A
0010.0400.0003  dynamic  Te0/0/0/10/0.1              N/A          Remotely Aged
N/A
2000.3000.4000  static   Te0/0/0/10/0.2              N/A          N/A
N/A

```

Verify the EVPN EVI information with the VPN-ID and MAC address filter.

```
RP/0/RSP0/CPU0:router# show evpn evi vpn-id 2001 mac
VPN-ID      Encap      MAC address  IP address  Nexthop
Label
-----
-----
2001                00a9.2002.0001 ::          10.10.10.10
34226    <-- Remote MAC
2001                00a9.2002.0001 ::          209.165.201.30
34202
2001                00a3.0001.0001 20.1.5.55    TenGigE0/6/0/1.36001
34203    <-- Local MAC

RP/0/RSP0/CPU0:router# RP/0/RSP0/CPU0:router# show evpn evi vpn-id 2001 mac 00a9.2002.0001
detail
EVI      MAC address      IP address  Nexthop      Label
----      -
2001     00a9.2002.0001  ::          10.10.10.10  34226
2001     00a9.2002.0001  ::          209.165.201.30  34202

Ethernet Tag : 0
Multi-paths Resolved : True <--- aliasing to two remote PE with All-Active load balancing

Static : No
Local Ethernet Segment : N/A
Remote Ethernet Segment : 0100.211b.fce5.df00.0b00
Local Sequence Number : N/A
Remote Sequence Number : 0
Local Encapsulation : N/A
Remote Encapsulation : MPLS
```

Verify the BGP routes associated with EVPN with bridge-domain filter.

```
RP/0/RSP0/CPU0:router# show bgp l2vpn evpn bridge-domain EVPN_2001 route-type 2
*> [2][0][48][00bb.2001.0001][0]/104
      0.0.0.0          0 i <----- locally learnt MAC
*>i[2][0][48][00a9.2002.00be][0]/104
      10.10.10.10 100 0 i <----- remotely learnt MAC
* i 209.165.201.30 100 0 i
```

EVPN Out of Service

The EVPN Out of Service feature enables you to control the state of bundle interfaces that are part of an Ethernet segment that have Link Aggregation Control protocol (LACP) configured. This feature enables you to put a node out of service (OOS) without having to manually shutdown all the bundles on their provider edge (PE).

Use the **cost-out** command to bring down all the bundle interfaces belonging to an Ethernet VPN (EVPN) Ethernet segment on a node. The Ethernet A-D Ethernet Segment (ES-EAD) routes are withdrawn before shutting down the bundles. The PE signals to the connected customer edge (CE) device to bring down the corresponding bundle member. This steers away traffic from this PE node without traffic disruption. The

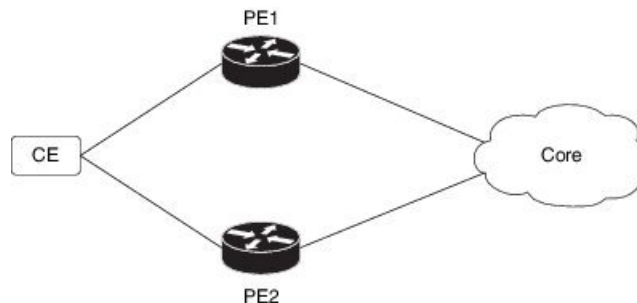
traffic that is bound for the Ethernet segment from the CE is directed to the peer PE in a multi-homing environment.



Note EVPN cost-out is supported only on manually configured ESIs.

In the following topology, the CE is connected to PE1 and PE2. When you configure the **cost-out** command on PE1, all the bundle interfaces on the Ethernet segment are brought down. Also, the corresponding bundle member is brought down on the CE. Hence, the traffic for this Ethernet segment is now sent to PE2 from the CE.

Figure 7: EVPN Out of Service



To bring up the node into service, use **no cost-out** command. This brings up all the bundle interfaces belonging to EVPN Ethernet segment on the PE and the corresponding bundle members on the CE.

When the node is in cost-out state, adding a new bundle Ethernet segment brings that bundle down. Similarly, removing the bundle Ethernet segment brings that bundle up.

Use **startup-cost-in** command to bring up the node into service after the specified time on reload. The node will cost-out when EVPN is initialized and remain cost-out until the set time. If you execute **evpn no startup-cost-in** command while timer is running, the timer stops and node is cost-in.

The 'cost-out' configuration always takes precedence over the 'startup-cost-in' timer. So, if you reload with both the configurations, cost-out state is controlled by the 'cost-out' configuration and the timer is not relevant. Similarly, if you reload with the startup timer, and configure 'cost-out' while timer is running, the timer is stopped and OOS state is controlled only by the 'cost-out' configuration.

If you do a proc restart while the startup-cost-in timer is running, the node remains in cost-out state and the timer restarts.

Configure EVPN Out of Service

This section describes how you can configure EVPN Out of Service.

```

/* Configuring node cost-out on a PE */

Router# configure
Router(config)# evpn
Router(config-evpn)# cost-out
Router(config-evpn)# commit

/* Bringing up the node into service */

Router# configure

```

```

Router(config)# evpn
Router(config-evpn)# no cost-out
Router(config-evpn)commit

/* Configuring the timer to bring up the node into service after the specified time on
reload */

Router# configure
Router(config)# evpn
Router(config-evpn)# startup-cost-in 6000
Router(config-evpn)commit

```

Running Configuration

```

configure
evpn
  cost-out
!

configure
evpn
  startup-cost-in 6000
!

```

Verification

Verify the EVPN Out of Service configuration.

```

/* Verify the node cost-out configuration */

Router# show evpn summary
Fri Apr 7 07:45:22.311 IST
Global Information
-----
Number of EVIs : 2
Number of Local EAD Entries : 0
Number of Remote EAD Entries : 0
Number of Local MAC Routes : 0
Number of Local MAC Routes : 5
      MAC : 5
      MAC-IPv4 : 0
      MAC-IPv6 : 0
Number of Local ES:Global MAC : 12
Number of Remote MAC Routes : 7
      MAC : 7
      MAC-IPv4 : 0
      MAC-IPv6 : 0
Number of Local IMCAST Routes : 56
Number of Remote IMCAST Routes: 56
Number of Internal Labels : 5
Number of ES Entries : 9
Number of Neighbor Entries : 1
EVPN Router ID : 192.168.0.1
BGP Router ID : ::
BGP ASN : 100
PBB BSA MAC address : 0207.1fee.be00
Global peering timer : 3 seconds
Global recovery timer : 30 seconds
EVPN cost-out : TRUE
      startup-cost-in timer : Not configured

```

```
/* Verify the no cost-out configuration */
```

```
Router# show evpn summary
Fri Apr 7 07:45:22.311 IST
Global Information
-----
Number of EVIs : 2
Number of Local EAD Entries : 0
Number of Remote EAD Entries : 0
Number of Local MAC Routes : 0
Number of Local MAC Routes : 5
      MAC : 5
      MAC-IPv4 : 0
      MAC-IPv6 : 0
Number of Local ES:Global MAC : 12
Number of Remote MAC Routes : 7
      MAC : 7
      MAC-IPv4 : 0
      MAC-IPv6 : 0
Number of Local IMCAST Routes : 56
Number of Remote IMCAST Routes: 56
Number of Internal Labels : 5
Number of ES Entries : 9
Number of Neighbor Entries : 1
EVPN Router ID : 192.168.0.1
BGP Router ID : ::
BGP ASN : 100
PBB BSA MAC address : 0207.1fee.be00
Global peering timer : 3 seconds
Global recovery timer : 30 seconds
EVPN cost-out : FALSE
      startup-cost-in timer : Not configured
```

```
/* Verify the startup-cost-in timer configuration */
```

```
Router# show evpn summary
Fri Apr 7 07:45:22.311 IST
Global Information
-----
Number of EVIs : 2
Number of Local EAD Entries : 0
Number of Remote EAD Entries : 0
Number of Local MAC Routes : 0
Number of Local MAC Routes : 5
      MAC : 5
      MAC-IPv4 : 0
      MAC-IPv6 : 0
Number of Local ES:Global MAC : 12
Number of Remote MAC Routes : 7
      MAC : 7
      MAC-IPv4 : 0
      MAC-IPv6 : 0
Number of Local IMCAST Routes : 56
Number of Remote IMCAST Routes: 56
Number of Internal Labels : 5
Number of ES Entries : 9
Number of Neighbor Entries : 1
EVPN Router ID : 192.168.0.1
BGP Router ID : ::
BGP ASN : 100
PBB BSA MAC address : 0207.1fee.be00
Global peering timer : 3 seconds
```

```

Global recovery timer      :      30 seconds
EVPN node cost-out        :      TRUE
    startup-cost-in timer :      6000

```

EVPN Multiple Services per Ethernet Segment

EVPN Multiple Services per Ethernet Segment feature allows you to configure multiple services over single Ethernet Segment (ES). Instead of configuring multiple services over multiple ES, you can configure multiple services over a single ES.

You can configure the following services on a single Ethernet Bundle; you can configure one service on each sub-interface.

- Flexible cross-connect (FXC) service. It supports VLAN Unaware, VLAN Aware, and Local Switching modes.

For more information, see *Configure Point-to-Point Layer 2 Services* chapter in *L2VPN and Ethernet Services Configuration Guide for Cisco NCS 5500 Series Routers*.

- EVPN-VPWS Xconnect service

For more information, see *EVPN Virtual Private Wire Service (VPWS)* chapter in *L2VPN and Ethernet Services Configuration Guide for Cisco NCS 5500 Series Routers*.

- EVPN Integrated Routing and Bridging (IRB)

For more information, see *Configure EVPN IRB* chapter in *L2VPN and Ethernet Services Configuration Guide for Cisco NCS 5500 Series Routers*.

- Native EVPN

For more information see, *EVPN Features* chapter in *L2VPN and Ethernet Services Configuration Guide for Cisco NCS 5500 Series Routers*.

All these services are supported only on all-active multihoming scenario.

Configure EVPN Multiple Services per Ethernet Segment

Consider a customer edge (CE) device connected to two provider edge (PE) devices through Ethernet Bundle interface 22001. Configure multiple services on Bundle Ethernet sub-interfaces.

Configuration Example

Consider Bundle-Ether22001 ES, and configure multiple services on sub-interface.

```

/* Configure attachment circuits */
Router# configure
Router(config)# interface Bundle-Ether22001.12 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 12
Router(config-l2vpn-subif)# exit
Router(config-l2vpn)# exit
Router(config)# interface Bundle-Ether22001.13 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 13
Router(config-l2vpn-subif)# exit
Router(config-l2vpn)# exit
Router(config)# interface Bundle-Ether22001.14 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 14
Router(config-l2vpn-subif)# exit

```

```

Router(config-l2vpn)# exit
Router(config)# interface Bundle-Ether22001.1 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 1
Router(config-l2vpn-subif)# exit
Router(config-l2vpn)# exit
Router(config)# interface Bundle-Ether22001.2 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 2
Router(config-l2vpn-subif)# exit
Router(config-l2vpn)# exit
Router(config)# interface Bundle-Ether22001.3 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 3
Router(config-l2vpn-subif)# exit
Router(config-l2vpn)# exit
Router(config)# interface Bundle-Ether22001.4 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 4
Router(config-l2vpn-subif)# exit
Router(config-l2vpn)# exit

/*Configure VLAN Unaware FXC Service */
Router(config)# l2vpn
Router(config-l2vpn)# flexible-xconnect-service vlan-unaware fxc_mh1
Router(config-l2vpn-fxs-vu)# interface Bundle-Ether22001.1
Router(config-l2vpn-fxs-vu)# interface Bundle-Ether22001.2
Router(config-l2vpn-fxs-vu)# interface Bundle-Ether22001.3
Router(config-l2vpn-fxs-vu)# neighbor evpn evi 21006 target 22016
Router(config-l2vpn-fxs-vu)# commit

/* Configure VLAN Aware FXC Service */
Router(config)# l2vpn
Router(config-l2vpn)# flexible-xconnect-service vlan-aware evi 24001
Router(config-l2vpn-fxs-va)# interface Bundle-Ether22001.12
Router(config-l2vpn-fxs-va)# interface Bundle-Ether22001.13
Router(config-l2vpn-fxs-va)# interface Bundle-Ether22001.14
Router(config-l2vpn-fxs-va)# commit

/* Configure Local Switching - Local switching is supported only on VLAN-aware FXC */
PE1
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# flexible-xconnect-service vlan-aware evi 31400
Router(config-l2vpn-fxs-va)# interface Bundle-Ether22001.1400
Router(config-l2vpn-fxs-va)# interface Bundle-Ether23001.1400
Router(config-l2vpn-fxs-va)# commit
Router(config-l2vpn-fxs)# exit
PE2
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# flexible-xconnect-service vlan-aware evi 31401
Router(config-l2vpn-fxs-va)# interface Bundle-Ether22001.1401
Router(config-l2vpn-fxs-va)# interface Bundle-Ether23001.1401
Router(config-l2vpn-fxs-va)# commit
Router(config-l2vpn-fxs)# exit

/* Configure EVPN-VPWS xconnect service and native EVPN with IRB */

Router# configure
Router(config)# interface Bundle-Ether22001.11 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 11
Router(config-l2vpn-subif)# rewrite ingress tag pop 2 symmetric
Router(config-l2vpn-subif)# commit
Router(config-l2vpn-subif)# exit

Router# configure

```

```

Router(config)# interface Bundle-Ether22001.21 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 1 second-dot1q 21
Router(config-l2vpn-subif)# rewrite ingress tag pop 2 symmetric
Router(config-l2vpn-subif)# commit
Router(config-l2vpn-subif)# exit

Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# xconnect group xg22001
Router(config-l2vpn-xc)# p2p evpn-vpws-mclag-22001
Router(config-l2vpn-xc-p2p)# interface Bundle-Ether22001.11
Router(config-l2vpn-xc-p2p)# neighbor evpn evi 22101 target 220101 source 220301
Router(config-l2vpn-xc-p2p-pw)# commit
Router(config-l2vpn-xc-p2p-pw)# exit

Router # configure
Router (config)# l2vpn
Router (config-l2vpn)# bridge group native_evpn1
Router (config-l2vpn-bg)# bridge-domain bd21
Router (config-l2vpn-bg-bd)# interface Bundle-Ether22001.21
Router (config-l2vpn-bg-bd-ac)# routed interface BVI21
Router (config-l2vpn-bg-bd-bvi)# evi 22021
Router (config-l2vpn-bg-bd-bvi)# commit
Router (config-l2vpn-bg-bd-bvi)# exit

/* Configure Native EVPN */
Router # configure
Router (config)# evpn
Router (config-evpn)# interface Bundle-Ether22001
Router (config-evpn-ac)# ethernet-segment identifier type 0 ff.ff.ff.ff.ff.ff.ff.ff.0e
Router (config-evpn-ac-es)# bgp route-target 2200.0001.0001
Router (config-evpn-ac-es)# exit
Router (config-evpn)# evi 24001
Router (config-evpn-evi)# bgp
Router (config-evpn-evi-bgp)# route-target import 64:24001
Router (config-evpn-evi-bgp)# route-target export 64:24001
Router (config-evpn-evi-bgp)# exit
Router (config-evpn-evi)# exit
Router (config-evpn)# evi 21006
Router (config-evpn-evi)# bgp
Router (config-evpn-evi-bgp)# route-target route-target 64:10000
Router (config-evpn-evi-bgp)# exit
Router (config-evpn-evi)# exit
Router (config-evpn)# evi 22101
Router (config-evpn-evi)# bgp
Router (config-evpn-evi-bgp)# route-target import 64:22101
Router (config-evpn-evi-bgp)# route-target export 64:22101
Router (config-evpn-evi-bgp)# exit
Router (config-evpn-evi)# exit
Router (config-evpn)# evi 22021
Router (config-evpn-evi)# bgp
Router (config-evpn-evi-bgp)# route-target import 64: 22021
Router (config-evpn-evi-bgp)# route-target export 64: 22021
Router (config-evpn-evi-bgp)# exit
Router (config-evpn-evi)# exit
Router (config-evpn-evi)# advertise-mac
Router (config-evpn-evi)# exit
Router (config-evpn)# evi 22022
Router (config-evpn-evi)# bgp
Router (config-evpn-evi-bgp)# route-target import 64: 22022
Router (config-evpn-evi-bgp)# route-target export 64: 22022
Router (config-evpn-evi-bgp)# exit

```



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

Running Configuration

```
/* Configure attachment circuits */
interface Bundle-Ether22001.12 l2transport
encapsulation dot1q 1 second-dot1q 12
!
interface Bundle-Ether22001.13 l2transport
encapsulation dot1q 1 second-dot1q 13
!
interface Bundle-Ether22001.14 l2transport
encapsulation dot1q 1 second-dot1q 14
!
interface Bundle-Ether22001.1 l2transport
encapsulation dot1q 1 second-dot1q 1
!
interface Bundle-Ether22001.2 l2transport
encapsulation dot1q 1 second-dot1q 2
!
interface Bundle-Ether22001.3 l2transport
encapsulation dot1q 1 second-dot1q 3
!
interface Bundle-Ether22001.4 l2transport
encapsulation dot1q 1 second-dot1q 4

/*Configure VLAN Unaware FXC Service */
flexible-xconnect-service vlan-unaware fxc_mh1
  interface Bundle-Ether22001.1
  interface Bundle-Ether22001.2
  interface Bundle-Ether22001.3
  neighbor evpn evi 21006 target 22016
!
/*Configure VLAN Aware FXC Service */
l2vpn
flexible-xconnect-service vlan-aware evi 24001
  interface Bundle-Ether22001.12
  interface Bundle-Ether22001.13
  interface Bundle-Ether22001.14

/* Configure Local Switching */
flexible-xconnect-service vlan-aware evi 31400
  interface Bundle-Ether22001.1400
  interface Bundle-Ether23001.1400
!
flexible-xconnect-service vlan-aware evi 31401
  interface Bundle-Ether22001.1401
  interface Bundle-Ether23001.1401
!

/* Configure EVPN-VPWS xconnect service and native EVPN with IRB */
interface Bundle-Ether22001.11 l2transport
  encapsulation dot1q 1 second-dot1q 11
  rewrite ingress tag pop 2 symmetric
!
interface Bundle-Ether22001.21 l2transport
  encapsulation dot1q 1 second-dot1q 21
  rewrite ingress tag pop 2 symmetric
!
!
```

```

l2vpn
xconnect group xg22001
p2p evpn-vpws-mclag-22001
interface Bundle-Ether22001.11
 neighbor evpn evi 22101 target 220101 source 220301
!
bridge group native_evpn1
 bridge-domain bd21
 interface Bundle-Ether22001.21
  routed interface BVI21
  evi 22021
!
/* Configure Native EVPN */
Evpn
interface Bundle-Ether22001
 ethernet-segment identifier type 0 ff.ff.ff.ff.ff.ff.ff.ff.ee
 bgp route-target 2200.0001.0001
!
 evi 24001
  bgp
   route-target import 64:24001
   route-target export 64:24001
!
 evi 21006
  bgp
   route-target 64:100006
!
 evi 22101
  bgp
   route-target import 64:22101
   route-target export 64:22101
!
 evi 22021
  bgp
   route-target import 64:22021
   route-target export 64:22021
!
 advertise-mac
!
 evi 22022
  bgp
   route-target import 64:22022
   route-target export 64:22022
!
 advertise-mac
!

```

Verification

Verify if each of the services is configured on the sub-interface.

```

Router# show l2vpn xconnect summary
Number of groups: 6
Number of xconnects: 505 Up: 505 Down: 0 Unresolved: 0 Partially-programmed: 0
AC-PW: 505 AC-AC: 0 PW-PW: 0 Monitor-Session-PW: 0
Number of Admin Down segments: 0
Number of MP2MP xconnects: 0
Up 0 Down 0
Advertised: 0 Non-Advertised: 0

```

```

Router# show l2vpn xconnect-service summary
Number of flexible xconnect services: 74

```

Up: 74

```
Router# show l2vpn flexible-xconnect-service name fxc_mh1
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
SB = Standby, SR = Standby Ready, (PP) = Partially Programmed
Flexible XConnect Service Segment
Name      ST  Type  Description  ST
-----
fxc_mh1  UP  AC:   BE22001.1   UP
          AC:   BE22001.2   UP
          AC:   BE22001.3   UP
-----
```

```
Router# show l2vpn flexible-xconnect-service evi 24001
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
SB = Standby, SR = Standby Ready, (PP) = Partially Programmed
Flexible XConnect Service Segment
Name      ST  Type  Description  ST
-----
evi:24001 UP  AC:   BE22001.11  UP
          AC:   BE22001.12  UP
          AC:   BE22001.13  UP
          AC:   BE22001.14  UP
-----
```

```
Router# show l2vpn xconnect group xg22001 xc-name evpn-vpws-mclag-22001
Fri Sep 1 17:28:58.259 UTC
Legend: ST = State, UP = Up, DN = Down, AD = Admin Down, UR = Unresolved,
SB = Standby, SR = Standby Ready, (PP) = Partially Programmed
XConnect
Group      Name                               ST      Segment 1      Segment 2
          Description ST      Description      Description      ST
-----
xg22001  evpn-vpws-mclag-22001  UP      BE22001.101  UP      EVPN 22101, 220101, 64.1.1.6  UP
-----
```

Associated Commands

- evpn
- evi
- ethernet-segment
- advertise-mac
- show evpn ethernet-segment
- show evpn evi
- show evpn summary
- show l2vpn xconnect summary
- show l2vpn flexible-xconnect-service
- show l2vpn xconnect group

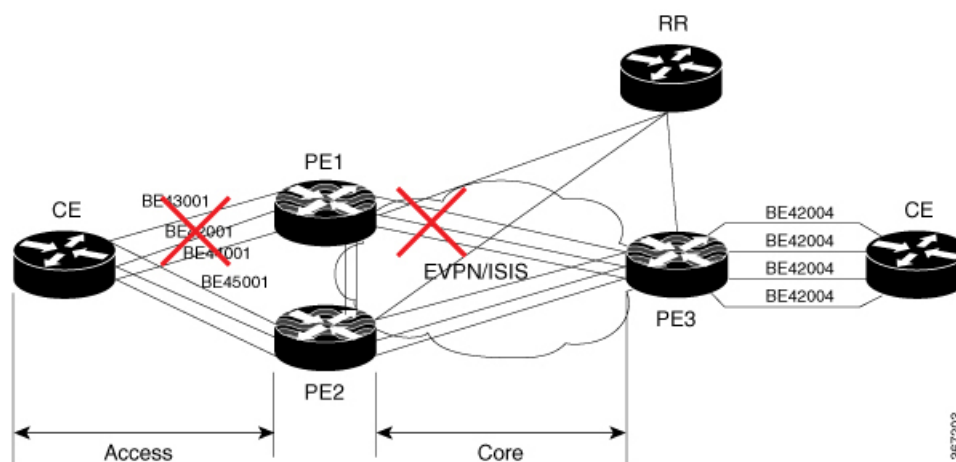
EVPN Core Isolation Protection

The EVPN Core Isolation Protection feature enables you to monitor and detect the link failure in the core. When a core link failure is detected in the provider edge (PE) device, EVPN brings down the PE's Ethernet Segment (ES), which is associated with access interface attached to the customer edge (CE) device.

EVPN replaces ICCP in detecting the core isolation. This new feature eliminates the use of ICCP in the EVPN environment.

Consider a topology where CE is connected to PE1 and PE2. PE1, PE2, and PE3 are running EVPN over the MPLS core network. The core interfaces can be Gigabit Ethernet or bundle interface.

Figure 8: EVPN Core Isolation Protection



When the core links of PE1 go down, the EVPN detects the link failure and isolates PE1 node from the core network by bringing down the access network. This prevents CE from sending any traffic to PE1. Since BGP session also goes down, the BGP invalidates all the routes that were advertised by the failed PE. This causes the remote PE2 and PE3 to update their next-hop path-list and the MAC routes in the L2FIB. PE2 becomes the forwarder for all the traffic, thus isolating PE1 from the core network.

When all the core interfaces and BGP sessions come up, PE1 advertises Ethernet A-D Ethernet Segment (ES-EAD) routes again, triggers the service carving and becomes part of the core network.

Configure EVPN Core Isolation Protection

Configure core interfaces under EVPN group and associate that group to the Ethernet Segment which is an attachment circuit (AC) attached to the CE. When all the core interfaces go down, EVPN brings down the associated access interfaces which prevents the CE device from using those links within their bundles. All interfaces that are part of a group go down, EVPN brings down the bundle and withdraws the ES-EAD route.

Starting from Cisco IOS-XR software version 7.1.2, you can configure a sub-interface as an EVPN Core. With this enhancement, when using IOS-XR software versions 7.1.2 and above, EVPN core facing interfaces can be physical, bundle main, or sub-interfaces. For all Cisco IOS-XR software versions lower than 7.1.2, EVPN core facing interfaces must be physical or bundle main. Sub-interfaces are not supported.

EVPN core facing interfaces can be physical main interface or subinterface, or bundle main interface or subinterface.

Restrictions

- A maximum of 24 groups can be created under the EVPN.
- A maximum of 12 core interfaces can be added under the group.
- The core interfaces can be reused among the groups. The core interface can be a bundle interface.
- EVPN group must only contain core interfaces, do not add access interfaces under the EVPN group.
- The access interface can only be a bundle interface.

```

Router# configure
Router(config)# evpn
Router(config-evpn)# group 42001
Router(config-evpn-group)# core interface GigabitEthernet0/2/0/1
Router(config-evpn-group)# core interface GigabitEthernet0/2/0/3
Router(config-evpn-group)#exit
!
Router(config-evpn)# group 43001
Router(config-evpn-group)# core interface GigabitEthernet0/2/0/2
Router(config-evpn-group)# core interface GigabitEthernet0/2/0/4
Router(config-evpn-group)#exit
!
Router# configure
Router(config)# evpn
Router(config-evpn)# interface bundle-Ether 42001
Router(config-evpn-ac)# core-isolation-group 42001
Router(config-evpn-ac)# exit
!
Router(config-evpn)# interface bundle-Ether 43001
Router(config-evpn-ac)# core-isolation-group 43001
Router(config-evpn-ac)# commit

```

Running Configuration

```

configure
evpn
  group 42001
    core interface GigabitEthernet0/2/0/1
    core interface GigabitEthernet0/2/0/3
    !
  group 43001
    core interface GigabitEthernet0/2/0/2
    core interface GigabitEthernet0/2/0/4
    !
!
configure
evpn
  interface bundle-Ether 42001
    core-isolation-group 42001
    !
  interface bundle-Ether 43001
    core-isolation-group 43001
    !
!

```

Verification

The **show evpn group** command displays the complete list of evpn groups, their associated core interfaces and access interfaces. The status, up or down, of each interface is displayed. For the access interface to be up, at least one of the core interfaces must be up.

```
Router# show evpn group /* Lists specific group with core-interfaces and access interface
status */
EVPN Group: 42001
  State: Ready
  Core Interfaces:
    Bundle-Ethernet110: down
    Bundle-Ethernet111: down
    GigabethEthernet0/2/0/1: up
    GigabethEthernet0/2/0/3: up
    GigabethEthernet0/4/0/8: up
    GigabethEthernet0/4/0/9: up
    GigabethEthernet0/4/0/10: up
  Access Interfaces:
    Bundle-Ether42001: up

EVPN Group: 43001
  State: Ready
  Core Interfaces:
    Bundle-Ethernet110: down
    GigabethEthernet0/2/0/2: up
    GigabethEthernet0/2/0/4: up
    GigabethEthernet0/4/0/9: up

  Access Interfaces:
    Bundle-Ether43001: up
```

Highest Random Weight Mode for EVPN DF Election

The Highest Random Weight (HRW) Mode for EVPN DF Election feature provides optimal load distribution of Designated Forwarder (DF) election, redundancy, and fast access. It ensures a nondisruptive service for an ES irrespective of the state of a peer DF.

The DF election is calculated based on the weight. The highest weight becomes the DF and the subsequent weight becomes a backup DF (BDF). The weight is determined by the mathematical function of EVI, ESI, and the IP address of the server.

DF weight calculation is based on the weight vector:

$$W_{rand}(v, S_i) = (1103515245((1103515245.S_i+12345)XOR D(v))+12345) \pmod{2^{31}}$$

where:

- S_i: IP Address of the server i
- v: EVI
- D(v): 31 bit digest [CRC-32 of v]

The existing DF election algorithm is based on ordinal value of a modulus calculation, and it comprises of number of peers and EVI. The DF is determined by the mathematical function of ESI and EVI, which is called “service carving”. This mode of DF election is described in RFC 7432.

In modulus calculation mode, the algorithm does not perform well when the Ethernet tags are all even or all odd. When the Ethernet Segment (ES) is multihomed to two PEs, all the VLANs pick only one of the PEs as the DF; one of the PEs does not get elected at all as the DF. The DF election is not optimal in this mode of operation.

The HRW mode of DF election has the following advantages over modulus mode of DF election:

- The DF election for the respective VLANs is equally distributed among the PEs.
- When a PE which is neither a DF nor a BDF hosts some VLANs on a given ES, and if the PE goes down, or its connection to the ES goes down, it does not result in a DF and BDF reassignment to the other PEs. This eliminates computation during the connection flaps.
- It avoids the service disruption that are inherent in the existing modulus based algorithm.
- The BDF provides redundant connectivity. The BDF ensures that there is no traffic disruption when a DF fails. When a DF fails, the BDF becomes the DF.

Configure Highest Random Weight Mode for EVPN DF Election

Perform this task to configure Highest Random Weight Mode for EVPN DF Election feature.

Configuration Example

```
Router# configure
Router(config)# evpn
Router(config-evpn)# interface Bundle-Ether 23
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# service-carving hrw
Router(config-evpn-ac-es)# commit
```

Running Configuration

```
configure
 evpn
   interface Bundle-Ether 23
     ethernet-segment
       service-carving hrw
   !
 !
 !
```

Verification

Verify that you have configured HRW mode of DF election.

```
Router#show evpn ethernet-segment interface bundleEther 23 carving detail
Ethernet Segment Id      Interface      Nexthops
-----
0011.1111.1111.1111.1111 Gi0/2/0/0    192.168.0.2
                        192.168.0.3

ES to BGP Gates      : Ready
ES to L2FIB Gates   : Ready
Main port           :
  Interface name     : GigabitEthernet0/2/0/0
  Interface MAC      : 02db.c740.ca4e
  IfHandle           : 0x01000060
  State              : Up
  Redundancy         : Not Defined
ESI type            : 0
  Value              : 11.1111.1111.1111.1111
ES Import RT        : 0011.0011.0011 (Local)
```

```

Source MAC      : 0000.0000.0000 (N/A)
Topology       :
  Operational   : MH, Single-active
  Configured    : Single-active (AAPS) (default)
Service Carving : HRW    -> Operation mode of carving
Peering Details : 192.168.0.2[HRW:P:00] 192.168.0.3[HRW:P:00] -> Carving capability as
advertised by peers
Service Carving Results:
  Forwarders    : 1
  Permanent     : 0
  Elected      : 0
  Not Elected  : 1
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 : 28109
Remote SHG labels : 1
                  24016 : nexthop 192.168.0.3

```

Associated Commands

- service-carving
- show evpn ethernet-segment

Network Convergence using Core Isolation Protection

The Network Convergence using Core Isolation Protection feature allows the router to converge fast when remote links and local interfaces fail. This feature reduces the duration of traffic drop by rapidly rerouting traffic to alternate paths. This feature uses Object Tracking (OT) to detect remote link failure and failure of connected interfaces.

Tracking interfaces can only detect failure of connected interfaces and not failure of a remote router interfaces that provides connectivity to the core. Tracking one or more BGP neighbor sessions along with one or more of the neighbor's address-families enables you to detect remote link failure.

Object Tracking

Object tracking (OT) is a mechanism for tracking an object to take any client action on another object as configured by the client. The object on which the client action is performed may not have any relationship to the tracked objects. The client actions are performed based on changes to the properties of the object being tracked.

You can identify each tracked object by a unique name that is specified by the track command in the configuration mode.

The tracking process receives the notification when the tracked object changes its state. The state of the tracked objects can be up or down.

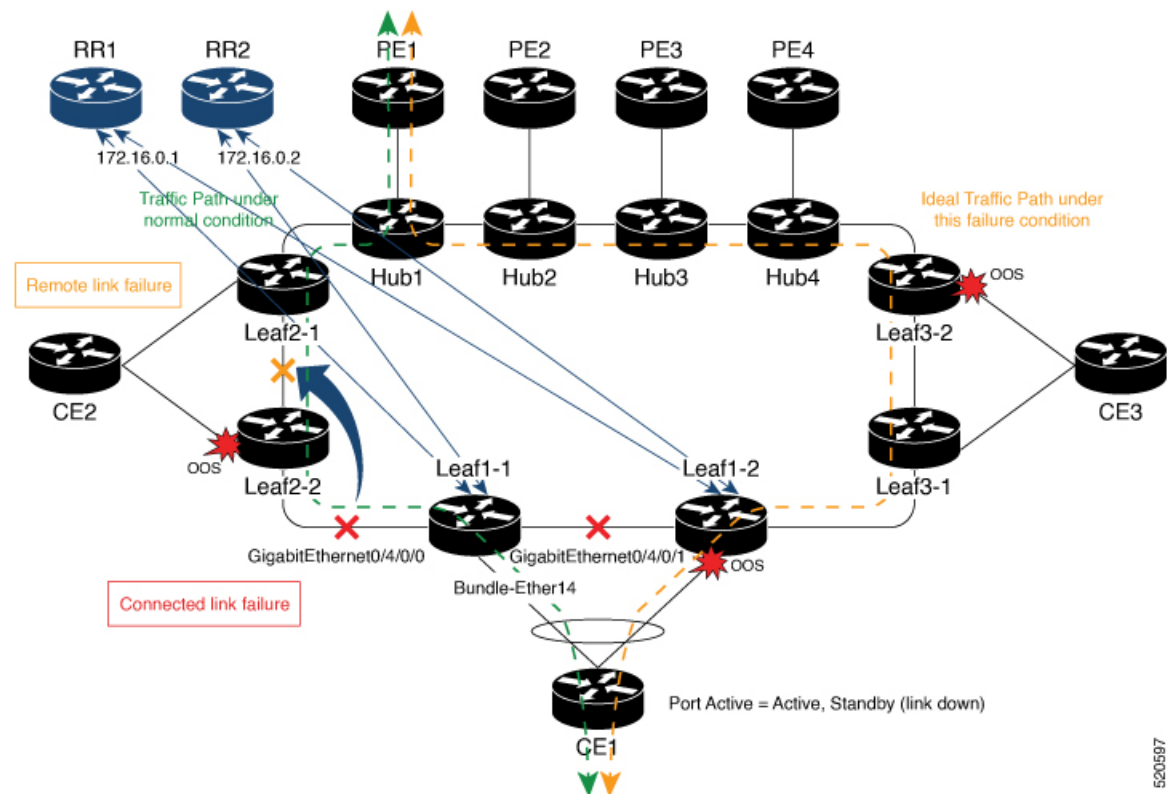
You can also track multiple objects by a list. You can use a flexible method for combining objects with Boolean logic. This functionality includes:

- Boolean AND function—When a tracked list has been assigned a Boolean AND function, each object defined within a subset must be in an up state, so that the tracked object can also be in the up state.

- Boolean OR function—When the tracked list has been assigned a Boolean OR function, it means that at least one object defined within a subset must also be in an up state, so that the tracked object can also be in the up state.

For more information on OT, see the *Configuring Object Tracking* chapter in the *System Management Configuration Guide for Cisco NCS 5500 Series Routers*.

Figure 9: EVPN Convergence Using Core Isolation Protection



Consider a traffic flow from CE1 to PE1. The CE1 can send the traffic either from Leaf1-1 or Leaf1-2. When Leaf1-1 loses the connectivity to both the local links and remote link, BGP sessions to both route reflectors (RRs) are down; the Leaf1-1 brings down the Bundle-Ether14 connected to CE1. The CE1 redirects the traffic from Leaf1-2 to PE1.

You can track the connected interfaces to identify the connected link failures. However, if there is a remote link failure, tracking connected interfaces does not identify the remote link failures. You must track BGP sessions to identify the remote link failure.



Note When you configure the **bgp graceful-restart** command, unconfiguring a neighbor is considered as a non-gr event. This generates a BGP notification to the neighbor before the neighbor is unconfigured.

On the remote router, if the track is configured for this neighbor, the track state is brought down immediately.

However, certain configurations are treated as graceful reset reason and when unconfigured they suppress the BGP notification to the neighbor. The route-reflector-client configuration under the neighbor or neighbor address-family is one of the examples.

On the remote router, if the track is configured for this neighbor, the track state is not brought down immediately because a notification is not received.

To overcome this situation, shutdown the neighbor before unconfiguring the neighbor. This generates a BGP notification to the neighbor, and any track configured for the neighbor is brought down immediately.

Configure EVPN Convergence using Core Isolation Protection

A tracked list contains one or more objects. The Boolean expression enables tracking objects using either AND or OR operators. For example, when tracking two interfaces, using the AND operator, up means that *both* interfaces are up, and down means that *either* interface is down.



Note An object must exist before it can be added to a tracked list.

The NOT operator is specified for one or more objects and negates the state of the object.

After configuring the tracked object, you must associate the neighbor or interface whose state must be tracked.

Perform the following tasks to configure EVPN convergence using core isolation protection:

- Configure BGP
- Track the Line Protocol State of an Interface
- Track neighbor address-family state
- Track objects for both interfaces and neighbors

Configuration Example

In this example, Leaf1-1 brings the down the AC connected to CE1 when:

Both local interfaces GigabitEthernet0/4/0/0 and GigabitEthernet0/4/0/1 of Leaf1-1 are down.

OR

Leaf1-1 BGP sessions to both RRs are down.

CE1 re-directs the traffic it was sending to Leaf1-1 to Leaf1-2.

Perform the following tasks on Leaf1-1:

```
/* Configure BGP */
Router# configure
Router(config)# router bgp 100
```

```

Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 172.16.0.1
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# neighbor 172.16.0.1
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# commit

/* Track the Line Protocol State of an Interface */
Router# configure
Router(config)# track interface-1
Router(config-track)# type line-protocol state
Router(config-track-line-prot)# interface GigabitEthernet0/4/0/0
Router(config-track-line-prot)#exit
Router(config-track)#exit
Router(config)# track interface-2
Router(config-track)# type line-protocol state
Router(config-track-line-prot)# interface GigabitEthernet0/4/0/1
Router(config-track-line-prot)#exit
Router(config-track)#exit
Router(config)# track interface-group-1
Router(config-track)# type list boolean or
Router(config-track-list-boolean)# object interface-1
Router(config-track-list-boolean)# object interface-2
Router(config-track-list-boolean)# commit

/* Track neighbor address-family state */
Router# configure
Router(config)# track neighbor-A
Router(config-track)# type bgp neighbor address-family state
Router(config-track-bgp-nbr-af)# address-family l2vpn evpn
Router(config-track-bgp-neighbor)# neighbor 172.16.0.1
Router(config-track-bgp-neighbor)# exit
Router(config-track-bgp-nbr-af)# exit
Router(config-track)# exit
Router(config)# track neighbor-B
Router(config-track)# type bgp neighbor address-family state
Router(config-track-bgp-nbr-af)# address-family l2vpn evpn
Router(config-track-bgp-neighbor)# neighbor 172.16.0.2
Router(config-track-bgp-neighbor)# exit
Router(config-track-bgp-nbr-af)# exit
Router(config-track)# exit
Router(config)# track neighbor-group-1
Router(config-track)# type list boolean or
Router(config-track-list-boolean)# object neighbor-A
Router(config-track-list-boolean)# object neighbor-B
Router(config-track-list-boolean)# commit

/* Track objects for both interfaces and neighbors */
Router# configure
Router(config)# track core-group-1
Router(config-track)# type list boolean and
Router(config-track-list-boolean)# object neighbor-group-1
Router(config-track-list-boolean)# object interface-group-1
Router(config-track-list-boolean)# action
Router(config-track-action)# track-down error-disable interface Bundle-Ether14 auto-recover
Router(config-track-action)# commit

```

Running Configuration

This section shows EVPN convergence using core isolation protection running configuration.

```

router bgp 100
  address-family l2vpn evpn
  !
  neighbor 172.16.0.1
    remote-as 100
    address-family l2vpn evpn
  !
  !
  neighbor 172.16.0.2
    remote-as 100
    address-family l2vpn evpn
  !
  !
!

track interface-1
  type line-protocol state
  interface GigabitEthernet0/4/0/0
  !
!

track interface-2
  type line-protocol state
  interface GigabitEthernet0/4/0/1
  !
!

track interface-group-1
  type list boolean or
  object interface-1
  object interface-2
  !
!

track neighbor-A
  type bgp neighbor address-family state
  address-family l2vpn evpn
  neighbor 172.16.0.1
  !
!

track neighbor-B
  type bgp neighbor address-family state
  address-family l2vpn evpn
  neighbor 172.16.0.1
  !
!

track neighbor-group-1
  type list boolean or
  object neighbor-A
  object neighbor-B
  !
!

track core-group-1
  type list boolean and
  object neighbor-group-1
  object interface-group-1
  !
action

```

```

track-down error-disable interface Bundle-Ether14 auto-recover
!
!

```

Verification

Verify that you have configured the EVPN convergence using core isolation protection feature successfully.

```

Router# show track
Wed May 27 04:42:11.995 UTC

```

```

Track neighbor-A
  BGP Neighbor AF L2VPN EVPN NBR 172.16.0.1 vrf default
  Reachability is UP
    Neighbor Address Reachability is Up
    BGP Neighbor Address-family state is Up
  4 changes, last change UTC Tue May 26 2020 20:14:33.171

```

```

Track neighbor-B
  BGP Neighbor AF L2VPN EVPN NBR 172.16.0.2 vrf default
  Reachability is UP
    Neighbor Address Reachability is Up
    BGP Neighbor Address-family state is Up
  4 changes, last change UTC Tue May 26 2020 20:14:27.527

```

```

Track core-group-1
  List boolean and is UP
  2 changes, last change 20:14:27 UTC Tue May 26 2020
  object interface-group-1 UP
  object neighbor-group-1 UP

```

```

Track interface-1
  Interface GigabitEthernet0/4/0/0 line-protocol
  Line protocol is UP
  2 changes, last change 20:13:32 UTC Tue May 26 2020

```

```

Track interface-2
  Interface GigabitEthernet0/4/0/1 line-protocol
  Line protocol is UP
  2 changes, last change 20:13:28 UTC Tue May 26 2020

```

```

Track interface-group-1
  List boolean or is UP
  2 changes, last change 20:13:28 UTC Tue May 26 2020
  object interface-2 UP
  object interface-1 UP

```

```

Track neighbor-group-1
  List boolean or is UP
  2 changes, last change 20:14:27 UTC Tue May 26 2020
  object neighbor-A UP
  object neighbor-B UP

```

```

Router# show track brief
Wed May 27 04:39:19.740 UTC

```

Track	Object	Parameter
Value		
neighbor-A	bgp nbr L2VPN EVPN 172.16.0.1 vrf defau	reachability
Up		
neighbor-B	bgp nbr L2VPN EVPN 172.16.0.1 vrf defau	reachability
Up		

```

core-group-1          list          boolean and
  Up
interface-1          interface GigabitEthernet0/4/0/0  line protocol
  Up
interface-2          interface GigabitEthernet0/4/0/1  line protocol
  Up
interface-group-1    list          boolean or
  Up
neighbor-group-1     list          boolean or
  Up

```

```
Router# show bgp track
```

```
Wed May 27 05:05:51.285 UTC
```

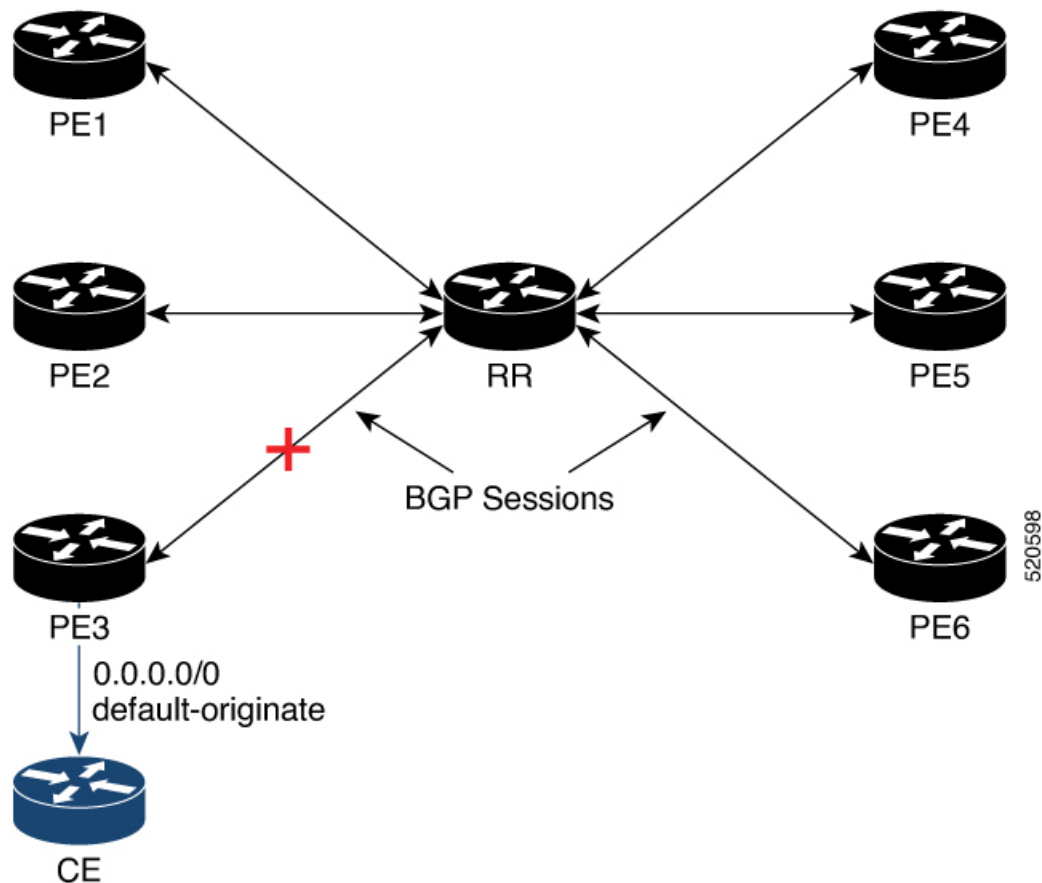
VRF	Address-family	Neighbor	Status	Flags
default	L2VPN EVPN	172.16.0.1	UP	0x01
default	L2VPN EVPN	172.16.0.2	UP	0x01

```
Processed 2 entries
```

Conditional Advertisement of Default-Originate

The router advertises the default-originate (0.0.0.0/0) towards the network fabric only upon receiving all the core routes. The router withdraws the advertisement of default-originate when the core is isolated. To avoid traffic drop, install the routes in the hardware. To accommodate an additional delay for the routes to be installed in the hardware, you can configure a timeout for the installed routes.

Figure 10: Advertisement of default-originate



In this topology, PE3 advertises the default-originate to CE only when the PE3 session to RR is established and all the routes are received from the RR.

Configure Conditional Advertisement of Default-Originate

Perform the following tasks to configure conditional advertisement of default-originate.

- Configure BGP
- Configure RPL
- Track BGP neighbor address-family state

Configuration Example

Perform the following task on PE3:

```
/* Configure BGP */
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# bgp router-id 192.0.2.1
Router(config-bgp)# address-family vpnv4 unicast
Router(config-bgp-af)# exit
```

```

Router(config-bgp)# neighbor 172.16.0.5
Router(config-bgp-nbr)# remote-as 200
Router(config-bgp-nbr)# address-family vpnv4 unicast
Router(config-bgp-nbr-af)# exit
Router(config-bgp-nbr)# exit
Router(config-bgp)# vrf cust1
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
Router(config-bgp-vrf)# neighbor 172.16.0.5
Router(config-bgp-vrf-nbr)# remote-as 200
Router(config-bgp-vrf-nbr)# address-family ipv4 unicast
Router(config-bgp-vrf-nbr-af)# default-originate route-policy track-bgp-core-policy
Router(config-bgp-vrf-nbr-af)# route-policy pass in
Router(config-bgp-vrf-nbr-af)# route-policy pass out
Router(config-bgp-vrf-nbr-af) commit

/* Configure RPL */
Router# configure
Router(config)# route-policy track-bgp-core-policy
Router(config-rpl)# if track core-group-1 is up then
Router(config-rpl-if)# pass
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
Router(config)# commit

/* Track BGP neighbor address-family state */
Router# configure
Router(config)# track core-group-1
Router(config-track)# type bgp neighbor address-family state
Router(config-track-bgp-nbr-af)# address-family vpnv4 unicast
Router(config-track-bgp-neighbor)# neighbor 172.16.0.5
Router(config-track-bgp-neighbor)# commit

```

Running Configuration

This section shows conditional advertisement of default-originate running configuration.

```

configure
router bgp 100
  bgp router-id 192.0.2.1
  address-family vpnv4 unicast
!
neighbor 172.16.0.5
  remote-as 200
  address-family vpnv4 unicast
!

vrf cust1
  rd auto
  address-family ipv4 unicast
  redistribute connected
  redistribute static
!
neighbor 172.16.0.5
  remote-as 200
  address-family ipv4 unicast
  default-originate route-policy track-bgp-core-policy
  route-policy pass in

```



```

    route-policy pass out
!

route-policy track-bgp-core-policy
  if track core-group-1 is up then
    pass
  endif
end-policy
!
track network-core
  type bgp neighbor address-family state
  address-family vpnv4 unicast
  neighbor 172.16.0.5
!

```

Verification

Verify conditional advertisement of default-originate.

```

Router# show rpl active route-policy
Wed May 27 06:54:31.902 UTC

```

```

ACTIVE -- Referenced by at least one policy which is attached
INACTIVE -- Only referenced by policies which are not attached
UNUSED -- Not attached (directly or indirectly) and not referenced

```

The following policies are (ACTIVE)

```

-----
      track-bgp-core
-----
Router# show rpl route-policy track-bgp-core-policy
Wed May 27 06:54:38.090 UTC
route-policy track-bgp-core-policy
  if track core-group-1 is up then
    pass
  endif
end-policy
!

```

```

Router# show bgp policy route-policy track-bgp-core-policy summary
Wed May 27 06:54:42.823 UTC
Network          Next Hop          From              Advertised to
0.0.0.0/0        0.0.0.0          Local             172.16.0.5

```

```

Router# show bgp neighbor 172.16.0.5
Wed May 27 06:55:39.535 UTC

```

```

BGP neighbor is 172.16.0.5
  Remote AS 9730, local AS 9730, internal link
  Remote router ID 172.16.0.5
  BGP state = Established, up for 10:41:12
[snip]
  For Address Family: IPv4 Unicast
  BGP neighbor version 2
  Update group: 0.4 Filter-group: 0.1 No Refresh request being processed
Default information originate: default route-policy track-bgp-core-policy, default sent
  AF-dependent capabilities:
[snip]
  Track Enabled, Status UP, Nbr GR state Not Enabled, EOR tmr Not Running
  Advertise routes with local-label via Unicast SAFI

```


- Connect hosts to leaf nodes. Leaf nodes are routed across the spines. For DC interconnectivity, the spines are connected through provider edge (PE) device and Data Center Interconnect (DCI).
- IS-IS labelled IGP and I-BGP are enabled internally across the leaf nodes, spine and DCI. The spine acts as a Route Reflector (RR).
- Configure IS-IS SR policy across the leaf node, spine and DCI.
- Configure BGP-LU between the DCs.
- Labelled Unicast BGP routes are learnt across the leaf nodes and tunnelled through IGP labelled paths (IS-IS SR).

For example, at Leaf428, BGP-LU routes are learnt for remote loopback 10.0.0.3 and 10.0.0.4.

- IRB (BVI) interface routes are learnt across the EVPN instances and programmed as labelled routes tunnelled through BGP-LU.

For example, at Leaf428, 192.0.2.1 can be reached with two BGP-LU paths 10.0.0.3 and 10.0.0.4.

Limitations

The following EVPN services are not supported over BGP-LU over IGP with L2 unicast and BUM traffic on the Cisco NCS 5500 series routers and NCS57 line cards:

- EVPN-ELAN and ELINE for EVPN Multi-Homing Single-Active
- EVPN IRB with VRF (intra-subnet)

Configure EVPN Bridging and VPWS Services over BGP-LU Underlay

Perform these tasks to configure the EVPN Bridging and VPWS Services over BGP-LU Underlay feature.

- Configure IGP
- Configure BGP
- Configure EVPN instance and ESI
- Configure BVI (IRB) Interface
- Configure VRF
- Configure BVI with VRF
- Configure VRF under BGP
- Configure bridge domain and associate with attachment circuits and EVPN instance
- Configure bridge domain and associate with attachment circuits, EVPN instance and BVI
- Configure EVPN VPWS

Configuration Example

```
/* Configure IGP */
IGP configuration is a pre-requisite to configure EVPN. IGP can be OSPF or ISIS.
Router# configure
```

```

Router(config)#router ospf 1
Router(config-ospf)#router-id 209.165.201.1
Router(config-ospf)#area 10
Router(config-ospf-ar)#interface loopback0\
Router(config-ospf-ar-if)#exit
Router(config-ospf-ar)#interface TenGigE0/0/0/1\
Router(config-ospf-ar-if)#exit
Router(config-ospf-ar)#interface TenGigE0/0/0/17\
Router(config-ospf-ar-if)#commit

/* Configure BGP */
Router# configure
Router(config)#router bgp 100
Router(config-bgp)#router-id 209.165.201.1
Router(config-bgp)#bgp graceful-restart
Router(config-bgp)#address-family ipv4 unicast
Router(config-bgp-af)#redistribute connected
Router(config-bgp-af)#network 209.165.200.225/27
Router(config-bgp-af)#allocate-label all
Router(config-bgp-af)#exit
Router(config-bgp)#address-family ipv6 unicast
Router(config-bgp-af)#allocate-label all
Router(config-bgp-af)#exit
Router(config-bgp)#neighbor-group spines
Router(config-bgp-nbrgrp)#remote-as 100
Router(config-bgp-nbrgrp)#update-source loopback0
Router(config-bgp-nbrgrp)#address-family ipv4 labeled-unicast multipath
Router(config-bgp-nbrgrp-af)#exit
Router(config-bgp-nbrgrp)#address-family ipv6 labeled-unicast multipath
Router(config-bgp-nbrgrp-af)#exit
Router(config-bgp-nbrgrp)#address-family l2vpn evpn
Router(config-bgp-nbrgrp-af)#advertise vpv4 unicast re-originated
Router(config-bgp-nbrgrp-af)#advertise vpv6 unicast re-originated
Router(config-bgp-nbrgrp-af)#exit
Router(config-bgp-nbrgrp)#exit
Router(config-bgp)#neighbor 209.165.200.225
Router(config-bgp-nbr)#use neighbor-group spines
Router(config-bgp-nbr)#commit

/* Configure VPN4 address-family */
Router(config)#router bgp 100
Router(config-bgp)#router-id 209.165.201.1
Router(config-bgp)#ibgp policy out enforce-modifications
Router(config-bgp)#address-family vpv4 unicast
Router(config-bgp-af)#commit

/* Configure EVPN instance and ESI */
Router#configure
Router(config)#evpn
Router(config-evpn)#evi 100
Router(config-evpn-instance)#advertise-mac
Router(config-evpn-instance-mac)#exit
Router(config-evpn-instance)#exit
Router(config-evpn)#interface Bundle-Ether1
Router(config-evpn-ac)#ethernet-segment identifier type 0 aa.aa.aa.aa.aa.aa.aa.aa.ac
Router(config-evpn-ac-es)#bgp route-target 0011.0011.0012
Router(config-evpn-ac)#commit

/* Configure BVI (IRB) Interface */
Router#configure
Router(config)#interface BVI200
Router(config-if)#ipv4 address 192.0.2.1 255.255.255.0

```

```

Router(config-if)#commit

/* Configure VRF */
Router# configure
Router(config)# vrf vpn2
Router(config-vrf)# address-family ipv4 unicast
Router(config-vrf-af)# import route-target 81:2
Router(config-vrf-af)# exit
Router(config-vrf)# address-family ipv6 unicast
Router(config-vrf-af)# import route-target 81:2
Router(config-vrf-af)# commit

/* Configure BVI with VRF */
Router(config)# interface BVI200
Router(config-if)# host-routing
Router(config-if)# vrf vpn72
Router(config-if-vrf)# ipv4 address ipv4 address 192.0.2.1 255.255.255.0
Router(config-if-vrf)# mac-address 10.1111.1
Router(config-if)# commit

/* Configure VRF under BGP */
Router(config)# router bgp 100
Router(config-bgp)# vrf vpn2
Router(config-bgp-vrf)# rd 102:2
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 8
Router(config-bgp-vrf-af)# redistribute connected
Router(config-bgp-vrf-af)# exit
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 8
Router(config-bgp-vrf-af)# redistribute connected
Router(config-bgp-vrf-af)# commit

/* Configure bridge domain and associate with attachment circuits and EVPN instance */
Router(config)#l2vpn
Router(config-l2vpn)#bridge group bg1
Router(config-l2vpn-bg)#bridge-domain bd1
Router(config-l2vpn-bg-bd)#interface BundleEther1.100
Router(config-l2vpn-bg-bd-ac)#evi 100
Router(config-l2vpn-bg-bd-evi)#commit

/* Configure bridge domain and associate with attachment circuits, EVPN instance and BVI */
Router(config)#l2vpn
Router(config-l2vpn)#bridge group bg2
Router(config-l2vpn-bg)#bridge-domain bd2
Router(config-l2vpn-bg-bd)#interface TenGigE0/0/0/38.200
Router(config-l2vpn-bg-bd-ac)#routed interface BVI200
Router(config-l2vpn-bg-bd-bvi)#evi 200
Router(config-l2vpn-bg-bd-bvi)#commit
Router(config-l2vpn-bg-bd-bvi)#exit

Router(config)#l2vpn
Router(config-l2vpn)#bridge group bg3
Router(config-l2vpn-bg)#bridge-domain bd3
Router(config-l2vpn-bg-bd)#interface TenGigE0/0/0/38.202
Router(config-l2vpn-bg-bd-ac)#routed interface BVI202
Router(config-l2vpn-bg-bd-bvi)#evi 202
Router(config-l2vpn-bg-bd-bvi)#commit

/* Configure EVPN VPWS */

```

```

Router#configure
Router(config)#router bgp 100
Router(config-bgp)#neighbor-group spines
Router(config-bgp-nbrgrp)#remote-as 100
Router(config-bgp-nbrgrp)#update-source loopback0
Router(config-bgp-nbrgrp)#address-family ipv4 labeled-unicast multipath
Router(config-bgp-nbrgrp-af)#exit
Router(config-bgp-nbrgrp)#address-family ipv6 labeled-unicast multipath
Router(config-bgp-nbrgrp-af)#exit
Router(config-bgp-nbrgrp)#address-family l2vpn evpn
Router(config-bgp-nbrgrp-af)#exit
Router(config-bgp-nbrgrp)#exit
Router(config-bgp)#neighbor 209.165.200.225
Router(config-bgp-nbr)#use neighbor-group spines
Router(config-bgp-nbr)#commit
Router(config-bgp-af)#exit
Router(config-bgp)#exit
Router(config)#l2vpn
Router(config-l2vpn)#xconnect group aa-evpn-vpws
Router(config-l2vpn-xc)#p2p vpws_513
Router(config-l2vpn-xc-p2p)#interface Bundle-Ether1.513
Router(config-l2vpn-xc-p2p)#neighbor evpn evi 513 target 513 source 513
Router(config-l2vpn-xc-p2p)# commit

```

Running Configuration

This section shows flooding disable running configuration.

```

/* Configure IGP */
router ospf 1
router-id 209.165.201.1
area 10
interface Loopback0
!
interface TenGigE0/0/0/1
!
interface TenGigE0/0/0/17
!
!
/* Configure BGP */
router bgp 100
router-id 209.165.201.1
bgp graceful-restart
address-family ipv4 unicast
redistribute connected
network 209.165.200.225/27
allocate-label all
address-family ipv6 unicast
allocate-label all
neighbor-group spines
remote-as 100
update-source loopback0
address-family ipv4 labeled-unicast multipath
!
address-family ipv6 labeled-unicast multipath
!
address-family l2vpn evpn
advertise vpv4 unicast re-originated
advertise vpv6 unicast re-originated
!
neighbor 209.165.200.225
use neighbor-group spines

```

```
!  
  
/* Configure VPN4 address-family */  
router bgp 100  
  router-id 209.165.201.1  
  ibgp policy out enforce-modifications  
  address-family vpnv4 unicast  
!  
  
/* Configure EVPN instance and ESI */  
evpn  
  evi 100  
    advertise-mac  
    !  
  interface Bundle-Ether1  
    ethernet-segment  
      identifier type 0 aa.aa.aa.aa.aa.aa.aa.aa.ac  
      bgp route-target 0011.0011.0012  
    !  
  !  
!  
  
/* Configuring BVI (IRB) Interface */  
configure  
  interface BVI200  
    ipv4 address 192.0.2.1 255.255.255.0  
  
/* Configure VRF */  
vrf vpn2  
  address-family ipv4 unicast  
    import route-target 81:2  
    !  
    !  
  !  
  address-family ipv6 unicast  
    import route-target 81:2  
    !  
    !  
  !  
  
/* Configure BVI with VRF */  
interface BVI200  
  host-routing  
  vrf vpn72  
    ipv4 address ipv4 address ipv4 address 192.0.2.1 255.255.255.0  
    mac-address 10.1111.1  
!  
  
/* Configure VRF under BGP */  
router bgp 100  
  vrf vpn2  
    rd 102:2  
    address-family ipv4 unicast  
      label mode per-vrf  
      maximum-paths ibgp 8  
      redistribute connected  
    !  
    address-family ipv6 unicast  
      label mode per-vrf  
      maximum-paths ibgp 8  
      redistribute connected  
    !  
  !  
!
```

```

/* Configure bridge domain and associate with attachment circuits and EVPN instance */
l2vpn
  bridge group bg1
    bridge-domain bd1
      interface Bundle-Ether1.100
      !
      evi 100

/*
bridge group bg2
  bridge-domain bd2
    interface TenGigE0/0/0/38.200
    !
    routed interface BVI200
    !
    evi 200
    !
    !

/* Configurige bridge domain and associate with attachment circuits, EVPN instance and BVI
*/
bridge group bg3
  bridge-domain bd3
    interface TenGigE0/0/0/38.202
    !
    routed interface BVI202
    !
    evi 202
    !
    !
    !

/* Configure EVPN VPWS */
configure
  router bgp 100
    neighbor-group spines
    remote-as 100
    update-source Loopback0
    address-family ipv4 labeled-unicast multipath
    !
    address-family ipv6 labeled-unicast multipath
    !
    address-family l2vpn evpn

neighbor 209.165.200.225
  use neighbor-group spines
  !
  !
l2vpn
  xconnect group aa-evpn-vpws
  p2p vpws_513
    interface Bundle-Ether1.513
    neighbor evpn evi 513 target 513 source 513

```

Verification

Verify that you have configured EVPN Bridging and VPWS Services over BGP-LU Underlay feature successfully.



Note Load Balancing is not supported for EVPN Bridging over BGP-LU with Multipaths.

```
Router#show cef vrf AIM9 10.0.0.1
Tue Jan 20 22:00:56.233 UTC
10.0.0.1/8, version 4, internal 0x5000001 0x0 (ptr 0x97d34b44) [1], 0x0 (0x0), 0x208
(0x98bef0f0)
Updated Mar 18 06:01:46.175
Prefix Len 32, traffic index 0, precedence n/a, priority 3
  via 10.0.0.3/8, 7 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 0 NHID 0x0 [0x972c6f08 0x0]
    recursion-via-/32
    next hop VRF - 'default', table - 0xe0000000
    next hop 10.0.0.3/8 via 16448/0/21
      next hop 192.0.2.1/24 BE128          labels imposed {16111 64013 80002}
  via 100.0.0.88/32, 7 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 1 NHID 0x0 [0x972c6d68 0x0]
    recursion-via-/32
    next hop VRF - 'default', table - 0xe0000000
    next hop 10.0.0.4/8 via 16488/0/21
      next hop 192.0.2.1/24 BE128          labels imposed {16111 64009 80002}
```

```
Router#show l2vpn xconnect group aa-evpn-vpws xc-name vpws_513 detail
Wed Jan 22 13:14:05.878 GMT+4

Group aa-evpn-vpws, XC vpws_513, state is up; Interworking none
AC: Bundle-Ether1.513, state is up
Type VLAN; Num Ranges: 1
Rewrite Tags: []
VLAN ranges: [513, 513]
MTU 1500; XC ID 0xa00005f7; interworking none
Statistics:
  packets: received 0, sent 0
  bytes: received 0, sent 0
  drops: illegal VLAN 0, illegal length 0
EVPN: neighbor 24000, PW ID: evi 513, ac-id 513, state is up ( established )
XC ID 0xc0000001
Encapsulation MPLS
Source address 209.165.200.225
Encap type Ethernet, control word enabled
Sequencing not set
LSP : Up
```

EVPN	Local	Remote
Label	29045	1048577
MTU	1500	1500
Control word	enabled	enabled
AC ID	513	513
EVPN type	Ethernet	Ethernet

```
Router# show evpn internal-label vpn-id 513 detail
Tue Jan 28 13:22:19.110 GMT+4
```

VPN-ID	Encap	Ethernet	Segment Id	EtherTag	Label
513	MPLS	0099.9900.0000.0000.9999		0	None
Multi-paths resolved: FALSE (Remote all-active)					
Multi-paths Internal label: None					
EAD/ES	10.0.0.5			0	

```

513 MPLS 0099.9900.0000.0000.9999 513 24000
Multi-paths resolved: TRUE (Remote all-active)
Multi-paths Internal label: 24000
EAD/ES 10.0.0.5 0
EAD/EVI (P) 10.0.0.5 29104
Summary pathlist:
0xffffffff (P) 10.0.0.5 29104
-----

```

Router# **show mpls forwarding labels 24000 hardware egress detail location 0/0/CPU0**

Tue Jan 28 13:22:19.110 GMT+4

```

Label Label or ID Interface Switched
-----
24000 29104 EVPN:513 10.0.0.5 N/A

```

Updated: Oct 18 13:14:02.193

Version: 137839, Priority: 3

Label Stack (Top -> Bottom): { 29104 }

NHID: 0x0, Encap-ID: 0x140ea00000002, Path idx: 0, Backup path idx: 0, Weight: 0

MAC/Encaps: 0/4, MTU: 0

Packets Switched: 0

LEAF - HAL pd context :
sub-type : MPLS, ecd_marked:0, has_collapsed_ldi:0
collapse_bwalk_required:0, ecdv2_marked:0,

HW Walk:

LEAF:

PI:0x308de88fb8 PD:0x308de89058 rev:5554240 type: MPLS (2)

LEAF location: LEM

FEC key: 0x23e0220000d71

label action: MPLS_NOP

LWLDI:

PI:0x309faa82c8 PD:0x309faa8308 rev:5554239 p-rev:5459825 5459825 ldi type:EOS0_EOS1

FEC key: 0x23e0220000d71 fec index: 0x0(0) num paths:2, bkup paths: 0

Collpased IMP LDI: ECD_MARKED

IMP pattern:3

PI:0x309faa82c8 PD:0x309faa8308 rev:5554239 p-rev:5459825 5459825

FEC key: 0x257c720000d71 fec index: 0x20000003(3) num paths:2

Path:0 fec index: 0x20018f14(102164) DSP fec index: 0x200001f8(504),

MPLS encap key: 0xf1b00000400140ea MPLS encap id: 0x400140ea Remote: 0

Label Stack: 29104 16012 dpa-rev:55458217

Path:1 fec index: 0x20018f15(102165) DSP fec index: 0x200001f9(505),

MPLS encap key: 0xf1b00000400140eb MPLS encap id: 0x400140eb Remote: 0

Label Stack: 29104 16012 dpa-rev:55458218

REC-SHLDI HAL PD context :

ecd_marked:10, collapse_bwalk_required:0, load_shared_lb:0

RSHLDI:

PI:0x3093d16af8 PD:0x3093d16bc8 rev:5494421 dpa-rev:36033167 flag:0x1

FEC key: 0x249e440000d71 fec index: 0x2001c169(115049) num paths: 1

p-rev:5459825

Path:0 fec index: 0x2001c169(115049) DSP fec index: 0x200001f8(504),

LEAF - HAL pd context :

sub-type : MPLS, ecd_marked:1, has_collapsed_ldi:0

collapse_bwalk_required:0, ecdv2_marked:0,

HW Walk:

LEAF:

PI:0x308de433b8 PD:0x308de43458 rev:5459864 type: MPLS (2)

```

LEAF location: LEM
FEC key: 0

LWLDI:
  PI:0x309ffe9798 PD:0x309ffe97d8 rev:5459825 p-rev:4927729 4927729 ldi
type:IMP_EOS0_EOS1
  FEC key: 0x1a1c740000d71 fec index: 0x0(0) num paths:2, bkup paths: 0
  IMP LDI: ECD_MARKED SERVICE_MARKED
  IMP pattern:3
  PI:0x309ffe9798 PD:0x309ffe97d8 rev:5459825 p-rev:4927729 4927729
  FEC key: 0x23e0220000d71 fec index: 0x20000002(2) num paths:2
  Path:0 fec index: 0x2001f8b4(129204) DSP fec index: 0x200001f8(504),
    MPLS encap key: 0xf1b0000040013ef0 MPLS encap id: 0x40013ef0 Remote: 0
    Label Stack: 16012 dpa-rev:35993054. <<< LU Label>>>
  Path:1 fec index: 0x2001f8b5(129205) DSP fec index: 0x200001f9(505),
    MPLS encap key: 0xf1b0000040013ef2 MPLS encap id: 0x40013ef2 Remote: 0
    Label Stack: 16012 dpa-rev:35993055 <<< LU Label>>>

REC-SHLDI HAL PD context :
ecd_marked:10, collapse_bwalk_required:0, load_shared_lb:0

RSHLDI:
  PI:0x308dd32c38 PD:0x308dd32d08 rev:4927729 dpa-rev:35005343 flag:0x3
  FEC key: 0x1a1c740000d71 fec index: 0x20000813(2067) num paths: 2
  p-rev:4926086
  Path:0 fec index: 0x2001eefd(126717) DSP fec index: 0x200001f8(504),
  Path:1 fec index: 0x2001eefe(126718) DSP fec index: 0x200001f9(505),
LEAF - HAL pd context :
  sub-type : MPLS, ecd_marked:1, has_collapsed_ldi:0
  collapse_bwalk_required:0, ecdv2_marked:0,
HW Walk:
LEAF:
  PI:0x308dde33b8 PD:0x308dde3458 rev:4924403 type: MPLS (2)
  LEAF location: LEM
  FEC key: 0

LWLDI:
  PI:0x308b04ea58 PD:0x308b04ea98 rev:4924400 p-rev:4924389 4924389 4924389 4924389
ldi type:IMP_EOS0_EOS1
  FEC key: 0x1a75340000d71 fec index: 0x0(0) num paths:4, bkup paths: 0
  IMP LDI: ECD_MARKED
  IMP pattern:3
  PI:0x308b04ea58 PD:0x308b04ea98 rev:4924400 p-rev:4924389 4924389 4924389 4924389

  FEC key: 0x1a74720000d71 fec index: 0x200001f8(504) num paths:4
  Path:0 fec index: 0x2001ee86(126598) DSP:0x21
    MPLS encap key: 0xf1b0000040015878 MPLS encap id: 0x40015878 Remote: 0
    Label Stack: 16005 dpa-rev:34999715
  Path:1 fec index: 0x2001ee87(126599) DSP:0x22
    MPLS encap key: 0xf1b000004001587a MPLS encap id: 0x4001587a Remote: 0
    Label Stack: 16005 dpa-rev:34999716
  Path:2 fec index: 0x2001ee88(126600) DSP:0xc000002
    MPLS encap key: 0xf1b0000040016980 MPLS encap id: 0x40016980 Remote: 0
    Label Stack: 16005 dpa-rev:34989935
  Path:3 fec index: 0x2001ee89(126601) DSP:0xc000003
    MPLS encap key: 0xf1b00000400157fc MPLS encap id: 0x400157fc Remote: 0
    Label Stack: 16005 dpa-rev:34989936

SHLDI:
  PI:0x30927740c8 PD:0x3092774198 rev:4924389 dpa-rev:34999705 flag:0x0
  FEC key: 0x1a75340000d71 fec index: 0x200001ff(511) num paths: 4 bkup paths: 0

  p-rev:4924311 4924329 8779 4920854
  Path:0 fec index: 0x2001ee8f(126607) DSP:0x21 Dest fec index: 0x0(0)

```

```

Path:1 fec index: 0x2001ee90(126608) DSP:0x22 Dest fec index: 0x0(0)
Path:2 fec index: 0x2001ee91(126609) DSP:0xc000002 Dest fec index: 0x0(0)
Path:3 fec index: 0x2001ee92(126610) DSP:0xc000003 Dest fec index: 0x0(0)
TX-NHINFO:
    PI: 0x308dc51298 PD: 0x308dc51318 rev:4924311 dpa-rev:34994174 Encap hdl:
0x3091632e98
    Encap id: 0x40010003 Remote: 0 L3 int: 1670 flags: 0x3
    npu_mask: 0x1 DMAC: 84:78:ac:2d:f8:1f
TX-NHINFO:
    PI: 0x308dc51c20 PD: 0x308dc51ca0 rev:4924329 dpa-rev:34994264 Encap hdl:
0x30916332c8
    Encap id: 0x40010001 Remote: 0 L3 int: 1679 flags: 0x3
    npu_mask: 0x1 DMAC: d4:6d:50:7c:f9:4d
TX-NHINFO:
    PI: 0x308dc51ff0 PD: 0x308dc52070 rev:8779 dpa-rev:61964 Encap hdl:
0x308e9f4980
    Encap id: 0x40010007 Remote: 0 L3 int: 1728 flags: 0x807
    npu_mask: 0x1 DMAC: 84:78:ac:2d:f8:22
TX-NHINFO:
    PI: 0x308dc51480 PD: 0x308dc51500 rev:4920854 dpa-rev:34989846 Encap hdl:
0x308e9f4db0
    Encap id: 0x40010005 Remote: 0 L3 int: 1727 flags: 0x807
    npu_mask: 0x1 DMAC: 40:55:39:11:37:39
LEAF - HAL pd context :
sub-type : MPLS, ecd_marked:1, has_collapsed_ldi:0
collapse_bwalk_required:0, ecdv2_marked:0,
HW Walk:
LEAF:
    PI:0x308dde35b8 PD:0x308dde3658 rev:4926089 type: MPLS (2)
    LEAF location: LEM
    FEC key: 0
LWLDI:
    PI:0x308b04eb48 PD:0x308b04eb88 rev:4926086 p-rev:4924389 4924389 4924389 4924389
ldi type:IMP_EOS0_EOS1
    FEC key: 0x1a75340000d71 fec index: 0x0(0) num paths:4, bkup paths: 0
    IMP LDI: ECD_MARKED
    IMP pattern:3
    PI:0x308b04eb48 PD:0x308b04eb88 rev:4926086 p-rev:4924389 4924389 4924389 4924389
    FEC key: 0x1a74820000d71 fec index: 0x200001f9(505) num paths:4
    Path:0 fec index: 0x2001ee81(126593) DSP:0x21
        MPLS encap key: 0xf1b000004001587c MPLS encap id: 0x4001587c Remote: 0
        Label Stack: 16006 dpa-rev:35002526
    Path:1 fec index: 0x2001ee82(126594) DSP:0x22
        MPLS encap key: 0xf1b000004001588a MPLS encap id: 0x4001588a Remote: 0
        Label Stack: 16006 dpa-rev:35002527
    Path:2 fec index: 0x2001ee83(126595) DSP:0xc000002
        MPLS encap key: 0xf1b0000040016964 MPLS encap id: 0x40016964 Remote: 0
        Label Stack: 16006 dpa-rev:34991843
    Path:3 fec index: 0x2001ee84(126596) DSP:0xc000003
        MPLS encap key: 0xf1b00000400157fe MPLS encap id: 0x400157fe Remote: 0
        Label Stack: 16006 dpa-rev:34991844
SHLDI:
    PI:0x30927740c8 PD:0x3092774198 rev:4924389 dpa-rev:34999705 flag:0x0
    FEC key: 0x1a75340000d71 fec index: 0x200001ff(511) num paths: 4 bkup paths: 0
    p-rev:4924311 4924329 8779 4920854
    Path:0 fec index: 0x2001ee8f(126607) DSP:0x21 Dest fec index: 0x0(0)
    Path:1 fec index: 0x2001ee90(126608) DSP:0x22 Dest fec index: 0x0(0)

```

```

Path:2 fec index: 0x2001ee91(126609) DSP:0xc000002 Dest fec index: 0x0(0)
Path:3 fec index: 0x2001ee92(126610) DSP:0xc000003 Dest fec index: 0x0(0)

TX-NHINFO:
  PI: 0x308dc51298 PD: 0x308dc51318 rev:4924311 dpa-rev:34994174 Encap hdl:
0x3091632e98
  Encap id: 0x40010003 Remote: 0 L3 int: 1670 flags: 0x3
  npu_mask: 0x1 DMAC: 84:78:ac:2d:f8:1f

TX-NHINFO:
  PI: 0x308dc51c20 PD: 0x308dc51ca0 rev:4924329 dpa-rev:34994264 Encap hdl:
0x30916332c8
  Encap id: 0x40010001 Remote: 0 L3 int: 1679 flags: 0x3
  npu_mask: 0x1 DMAC: d4:6d:50:7c:f9:4d

TX-NHINFO:
  PI: 0x308dc51ff0 PD: 0x308dc52070 rev:8779 dpa-rev:61964 Encap hdl:
0x308e9f4980
  Encap id: 0x40010007 Remote: 0 L3 int: 1728 flags: 0x807
  npu_mask: 0x1 DMAC: 84:78:ac:2d:f8:22

TX-NHINFO:
  PI: 0x308dc51480 PD: 0x308dc51500 rev:4920854 dpa-rev:34989846 Encap hdl:
0x308e9f4db0
  Encap id: 0x40010005 Remote: 0 L3 int: 1727 flags: 0x807
  npu_mask: 0x1 DMAC: 40:55:39:11:37:39

```

Related Topics

[EVPN Bridging and VPWS Services over BGP-LU Underlay, on page 58](#)

Associated Commands

- show l2vpn bridge-domain
- show bgp l2vpn evpn neighbors
- show cef vrf

Support for DHCPv4 and DHCPv6 Client over BVI

The Support for DHCPv4 and DHCPv6 Client over the BVI feature allows you to configure DHCPv4 and DHCPv6 client on the Bridged Virtual Interface (BVI). You can configure a BVI, and request DHCP IPv4 or IPv6 address on the BVI. This allows your customer's device to have initial connectivity to your network without user intervention in the field. After the device is connected to your network, the customer devices can push a node-specific configuration with static IP addresses on a different BVI for customer deployment.

Configure DHCPv4 and DHCPv6 Client over BVI

Perform the following tasks to configure DHCPv4 and DHCPv6 client over BVI:

- Configure AC interface
- Configure L2VPN
- Configure BVI

Configuration Example

```

/* Configure AC interface */
Router# configure
Router(config)# interface tenGigE 0/5/0/1/1
Router(config-if)# bundle id 1 mode on
Router(config-if)# exit
Router(config)# interface Bundle-Ether1
Router(config-if)# no shut
Router(config-if)# exit
Router(config)# interface bundle-ether 1.100 l2transport
Router(config-l2vpn-subif)# encapsulation dot1q 100
Router(config-l2vpn-subif)# rewrite ingress tag pop 1 symmetric
Router(config-l2vpn-subif)# commit

/* Configure L2VPN */
Router # configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group BVI
Router(config-l2vpn-bg)# bridge-domain bvi
Router(config-l2vpn-bg-bd)# interface Bundle-Ether1.100
Router(config-l2vpn-bg-bd-ac)#exit
Router(config-l2vpn-bg-bd)# routed interface BVII1
Router(config-l2vpn-bg-bd-bvi)# commit

/* Configure BVI */
Router# configure
Router(config)# interface BVII1
Router(config-if)# ipv4 address dhcp
Router(config-if)# ipv6 address dhcp
Router(config-if)# commit

```

Running Configuration

This section shows the DHCPv4 and DHCPv6 client over BVI running configuration.

```

interface TenGigE0/5/0/1/1
bundle id 1 mode on
!
interface Bundle-Ether1
!
interface Bundle-Ether1.100 l2transport
encapsulation dot1q 100
rewrite ingress tag pop 1 symmetric
!
l2vpn
bridge group BVI
  bridge-domain bvi
    interface Bundle-Ether1.100
    !
    routed interface BVII1
    !
  !
!
interface BVII1
ipv4 address dhcp
ipv6 address dhcp
!

```

Verification

The show output given in the following section display the details of DHCPv4 and DHCPv6 client over BVI configuration.

```
Router# show l2vpn bridge-domain
Legend: pp = Partially Programmed.
Bridge group: BVI, bridge-domain: bvi, id: 0, state: up, ShgId: 0, MSTi: 0
  Aging: 300 s, MAC limit: 64000, Action: none, Notification: syslog
  Filter MAC addresses: 0
  ACs: 2 (2 up), VFIs: 0, PWs: 0 (0 up), PBBs: 0 (0 up), VNIs: 0 (0 up)
  List of ACs:
    BVI1, state: up, BVI MAC addresses: 1
    BE1.100, state: up, Static MAC addresses: 0
  List of Access PWs:
  List of VFIs:
  List of Access VFIs:
```

```
Router# show dhcp ipv4 client
```

Interface name	IP Address	Binding State	Lease Time Rem
BVI1	172.16.0.2	BOUND	3598 secs (00:59:58)

```
Router# show dhcp ipv6 client
```

Interface name	IPv6 Address	State	Lease Time Rem
BVI1	2000::1	BOUND	2591982

```
Router# show dhcp ipv4 client bvi1 detail
```

```
-----
Client Interface name      : BVI1
Client Interface handle    : 0x8804054
Client ChAddr             : 008a.9628.ac8a
Client ID                  : BVI1.00:8a:96:28:ac:8a
Client State               : BOUND
Client IPv4 Address (Dhcp) : 172.16.0.2
Client IPv4 Address Mask   : 255.240.0.0
Client Lease Time Allocated : 3600 secs (01:00:00)
Client Lease Time Remaining : 3571 secs (00:59:31)
Client Selected Server Address: 172.16.0.1
Client Next Hop Address    : 0.0.0.0
-----
```

```
Router# show dhcp ipv4 client BVI1 statistics
```

```
Client Interface name      : BVI1
-----
CLIENT COUNTER(s)        | VALUE
-----
Num discovers sent        :      44
Num requests sent         :       1
Num offers received       :       1
Num acks received         :       1
-----
```

```
Router# show dhcp ipv6 client
```

Interface name	IPv6 Address	State	Lease Time Rem
----------------	--------------	-------	----------------

```
BVI1                2000::1                BOUND                2591685
```

```
Router# show dhcp ipv6 client statistics-all
```

```
Interface name      : BVI1
Interface handle    : 0x8804054
VRF                 : 0x60000000
```

TYPE	TRANSMIT	RECEIVE	DROP
SOLICIT	17	0	0
ADVERTISE	0	1	0
REQUEST	1	0	0
REPLY	0	2	0
CONFIRM	0	0	0
RENEW	1	0	0
REBIND	0	0	0
RELEASE	0	0	0
RECONFIG	0	0	0
INFORM	0	0	0

TIMER	STARTED	STOPPED	EXPIRED
INIT	1	0	1
VBIND	0	0	0
RENEW	2	1	0
REBIND	2	1	0
RETRANS	19	3	16
VALID	2	1	0

Configure DHCPv6 Client Options

You can configure different DHCPv6 client options to differentiate between clients as required. Configure different DHCPv6 client options to differentiate how a DHCPv6 client communicates with a DHCPv6 server. The different DHCPv6 client options that you can configure are:

- **DUID:** If the DUID DHCPv6 client option is configured on an interface, DHCPv6 client communicates with the DHCPv6 server through the link layer address.
- **Rapid Commit:** If the Rapid Commit DHCPv6 client option is configured on an interface, DHCPv6 client can obtain configuration parameters from the DHCPv6 server through a rapid two-step exchange (solicit and reply) instead of the default four-step exchange (solicit, advertise, request, and reply).
- **DHCP Options:** The various other DHCPv6 options that can be configured on a DHCPv6 client are:
 - **Option 15:** Option 15 is also known as the User Class option and it is used by a DHCPv6 client to identify the type or category of users or applications it represents.
 - **Option 16:** Option 16 is also known as the Vendor ID option and it is used by a DHCPv6 a client to identify the vendor that manufactured the hardware on which the client is running.
 - **Option 23:** Option 23 is also known as the Domain name Server (DNS) option provides a list of one or more IPv6 addresses of DNS recursive name servers to which a client's DNS resolver can send DNS queries.
 - **Option 24:** Option 24 is also known as the Domain List option and it specifies the domain search list that the client uses to resolve hostnames with the DNS.

- **DHCP Timers:** This option is used to set different timer value for DHCP client configurations. The various DHCP timer options are:
 - **Release-timeout:** It is used to set retransmission timeout value for the initial release message.
 - **Req-max-rt:** It is used to set the maximum retransmission timeout value for the request message.
 - **Req-timeout:** It is used to set the initial request timeout value of the request message.
 - **Sol-max-delay:** It is used to set the maximum delay time of the first solicit message.
 - **Sol-max-rt:** It is used to set the maximum solicit retransmission time.
 - **Sol-time-out:** It is used to set the initial timeout value of the solicit message.

Configuration Example

Perform this task to configure DHCPv6 client options on a BVI interface.

```
Router# configure
Router(config)# interface BVI 10
Router(config-if)# ipv6 address dhcp-client-options
Router(config-dhcpv6-client)# duid linked-layer-address
Router(config-dhcpv6-client)# rapid-commit
Router(config-dhcpv6-client)# timers release-timeout 3
Router(config-dhcpv6-client)# timers sol-max-delay 1
Router(config-dhcpv6-client)# timers sol-time-out 1
Router(config-dhcpv6-client)# timers sol-max-rt 120
Router(config-dhcpv6-client)# timers req-max-rt 30
Router(config-dhcpv6-client)# timers req-timeout 1
Router(config-dhcpv6-client)# commit
```

Verification

To verify the DHCPv6 client options, use the **show dhcp ipv6 client BVI10 detail** command.

```
Router# show dhcp ipv6 client BVI10 detail
Wed Jun 10 16:19:21.272 IST

-----
Client Interface name : MgmtEth0/0/CPU0/1
Client Interface handle : 0x4040
Client MACAddr : 02f0.2b39.44be
Client State : BOUND
Client Link Local Address : fe80::f0:2bff:fe39:44be
Client IPv6 Address (Dhcp) : 600:1::12
Lease Remaining (in secs) : 74
DUID : 0003000102f02b3944be

Client Configuration
Timers
SOL_MAX_DELAY : 1 secs (00:00:01)
SOL_TIMEOUT : 1 secs (00:00:01)
SOL_MAX_RT : 120 secs (00:02:00)
REQ_TIMEOUT : 1 secs (00:00:01)
REQ_MAX_RT : 30 secs (00:00:30)
REL_TIMEOUT : 3 secs (00:00:01)

Options
RAPID-COMMIT : True
USER-CLASS : ciscoupnp
```

```

VENDOR-CLASS : vendor
DNS-SERVERS : True
DOMAIN-LIST : True

DUID Type : DUID_LL

Server Information
Server Address : fe80::d2:a1ff:feb2:3b9f
Preference : 0
DUID : 000300010206826e2e00
Status : SUCCESS
IA-NA
Status : SUCCESS
IAID : 0x40400001
T1 : 60 secs (00:01:00)
T2 : 96 secs (00:01:36)
IA-ADDR
IA NA Address : 600:1::12
Preferred Time : 120 secs (00:02:00)
Valid Time : 120 secs (00:02:00)
Flags : 0x0

```

Related Topics

- [Support for DHCPv4 and DHCPv6 Client over BVI, on page 69](#)

Associated Commands

- `show l2vpn bridge-domain`
- `show dhcp ipv4 client`
- `show dhcp ipv6 client`
- `show dhcp ipv4 client bvi`

EVPN Access-Driven DF Election

This feature includes a preference-based and access-driven DF election mechanism.

In a preference-based DF election mechanism, the weight decides which PE is the DF at any given time. You can use this method for topologies where interface failures are revertive. However, for topologies where an access-PE is directly connected to the core PE, use the access-driven DF election mechanism.

When access PEs are configured in a non-revertive mode, the access-driven DF election mechanism allows the access-PE to choose which PE is the DF.

Consider an interface in an access network that connects PE nodes running Multichassis Link Aggregation Control Protocol (mLACP) and the EVPN PE in the core. When this interface fails, there may be a traffic loss for a longer duration. The delay in convergence is because the backup PE is not chosen before failure occurs.

The EVPN Access-Driven DF Election feature allows the EVPN PE to preprogram a backup PE even before the failure of the interface. In the event of failure, the PE node will be aware of the next PE that will take over. Thereby reducing the convergence time. Use the *preference df weight* option for an Ethernet segment identifier (ESI) to set the backup path. By configuring the weight for a PE, you can control the DF election, thus define the backup path.

Restrictions

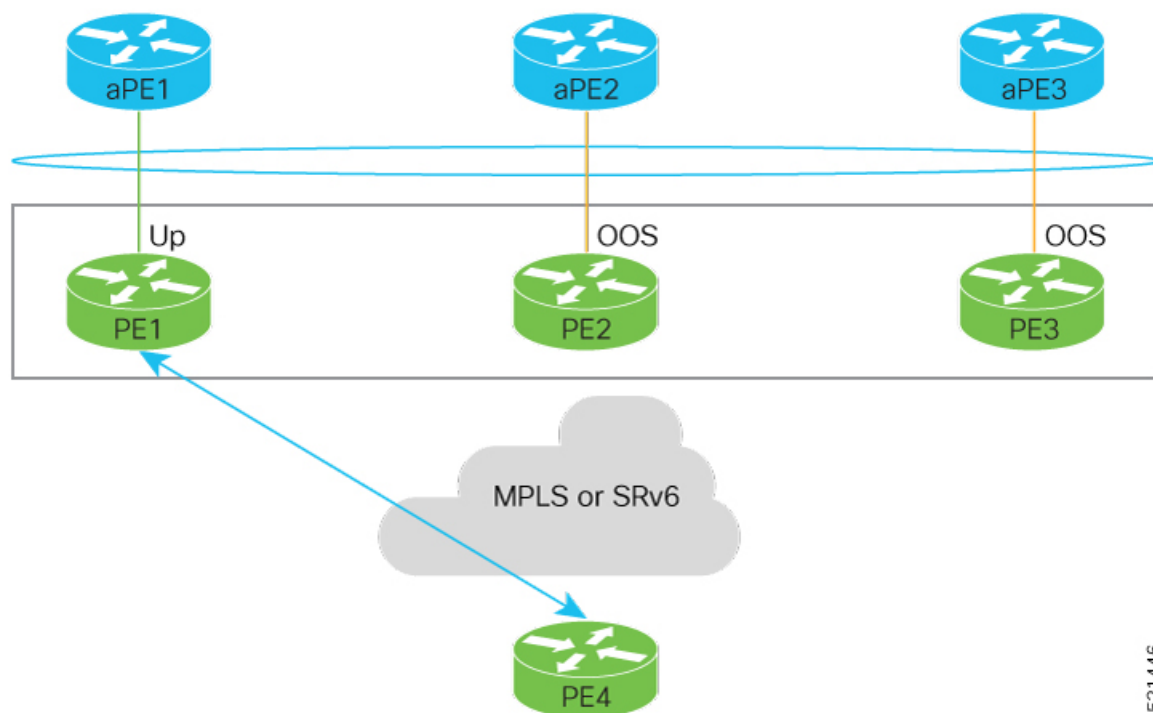
- The feature is supported only in an EVPN-VPWS scenario where EVPN PEs are in the port-active mode.
- The bundle attached to the ethernet segment must be configured with **lACP mode active**.

LACP mode on is not supported.

Topology

Let's understand the feature on how the backup path is precomputed with the following topology.

Figure 12: EVPN Access-Driven DF Election

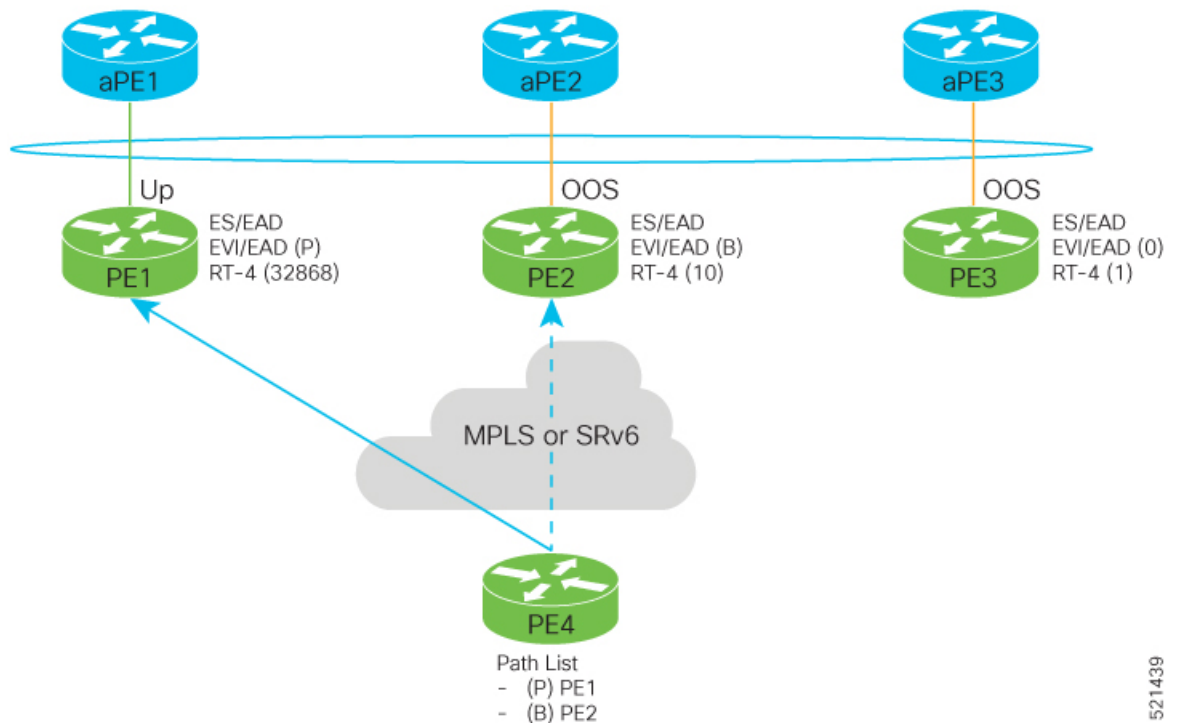


521446

- PE1, PE2, and PE3 are PEs for the EVPN core network.
- aPE1, aPE2, and aPE3 are their access PE counterparts and configured in a multichassis link aggregation group (MCLAG) redundancy group. Only one link among the three is active at any given time. aPE1, aPE2, and aPE3 are in a non-revertive mode.
- PE1 is directly connected to aPE1, PE2 to aPE2, and PE3 to aPE3. EVPN VPWS is configured on the PE devices in the core.
- All PE devices are attached to the same bundle and shares the same ethernet segment identifier.
- PE1, PE2, and PE3 are configured with a weight of 100, 10, and 1 respectively.

Traffic Flow

In this example, consider a traffic flow from a host connected to PE4 to the host connected to the access PE.



521439

- aPE1-PE1 interface state is up. The aPE2-PE2 and aPE3-PE3 remains in OOS state.
- The traffic is sent from PE4 to aPE1 through PE1 as the PE1 is configured with a highest weight of 100.
- The highest weight is modified by adding 32768 to the configured weight. For example, the weight of PE1 is 100, 32768 is added to this weight. Hence, 32868 is advertised to the peer PEs.
- The highest weight is advertised as P-bit, which is primary. The next highest weight is advertised as B-bit, which is secondary. The lowest weight as non-DF (NDF).
- When the EVPN PE devices are of same weight, the traffic is sent based on the IP address. Lowest IP address takes the precedence.
- Only one PE indicates that the state of the bundle for the Ethernet Segment is up. For all other PEs, the Ethernet Segment is standby and the bundle is in OOS state.
- All PE devices are aware of the associated next hop and weights of their peers.

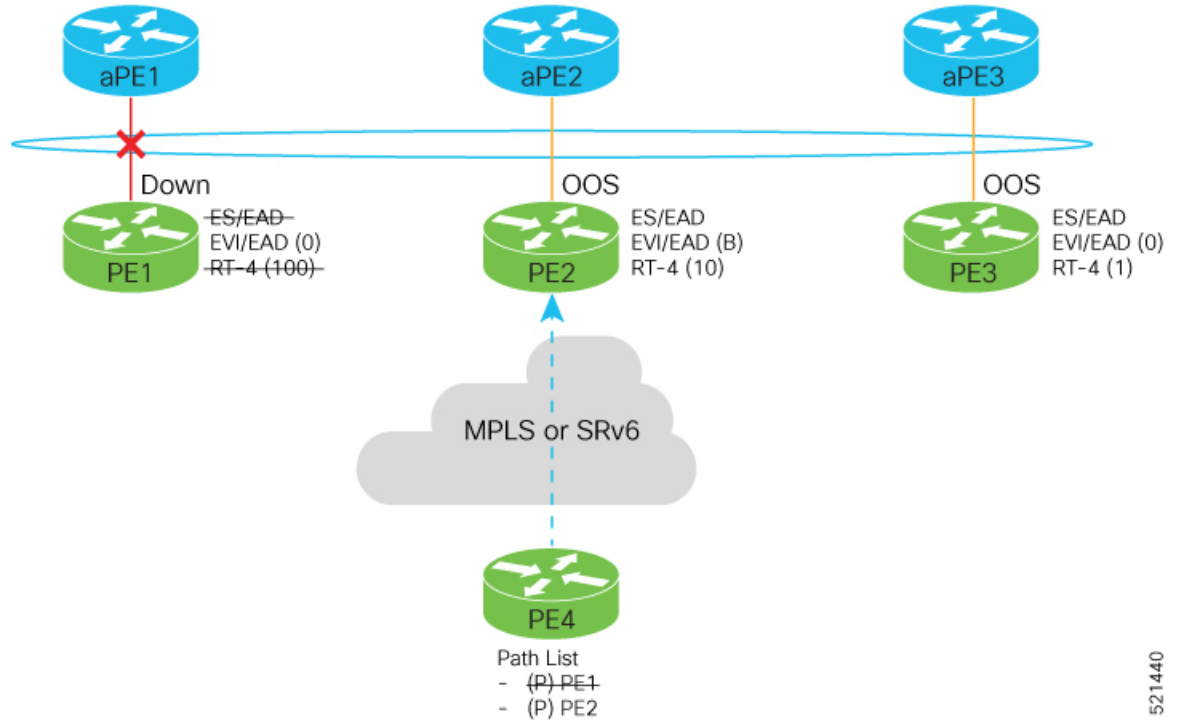
Failure and Recovery Scenarios

The weights configured on the EVPN PE devices cascade in the same order as the protection mechanism on the access side PEs:

- During the network failure, the redundancy ordering for the access PEs is aPE1, aPE2, aPE3.
- The weights of PE1 through PE3 are weight of PE1 > weight of PE2 > weight of PE3.
- If this ordering is not satisfied, the network will eventually converge, but it will not be as efficient as if the weights are ordered correctly.

Scenario - 1

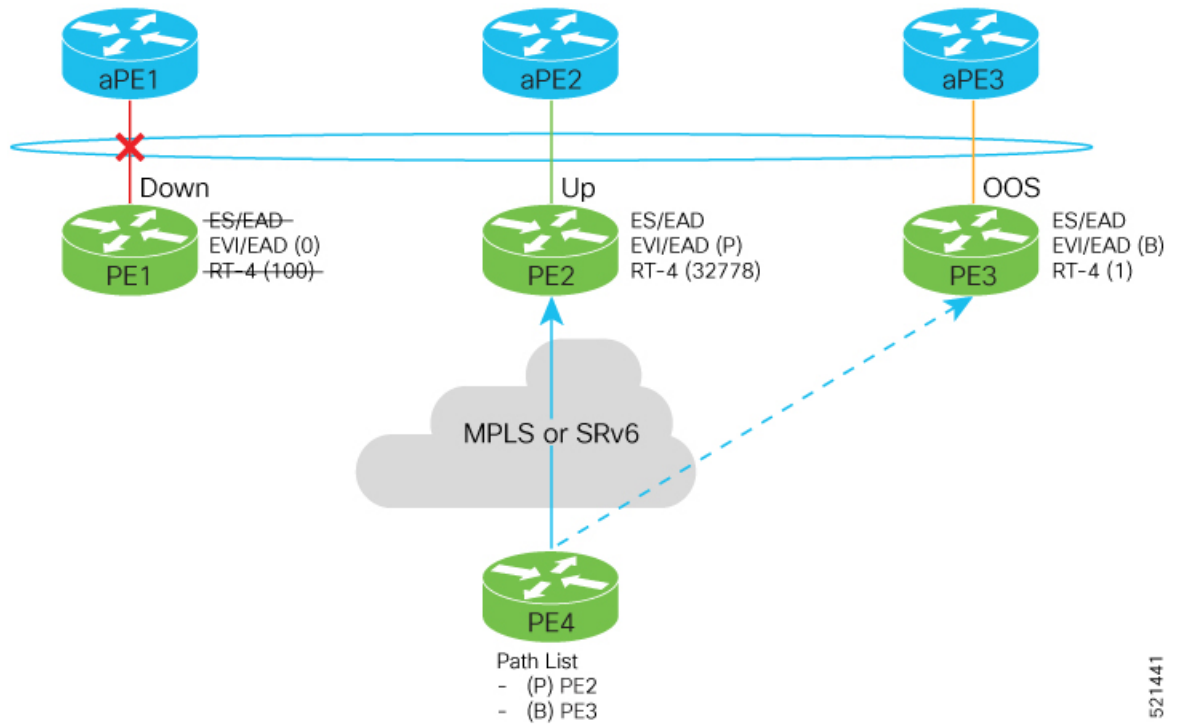
Consider a scenario where the aPE1-PE1 interface is down.



When aPE1-PE1 interface is down, the PE1 withdraws the EAD/ES route, and the traffic is sent through the backup path, which is PE2.

The aPE2-PE2 becomes the primary with a weight of 32778, and aPE3-PE3 becomes the backup. The aPE2-PE2 advertises P-bit to PE4. aPE3-PE3 advertises the B-bit to PE4.

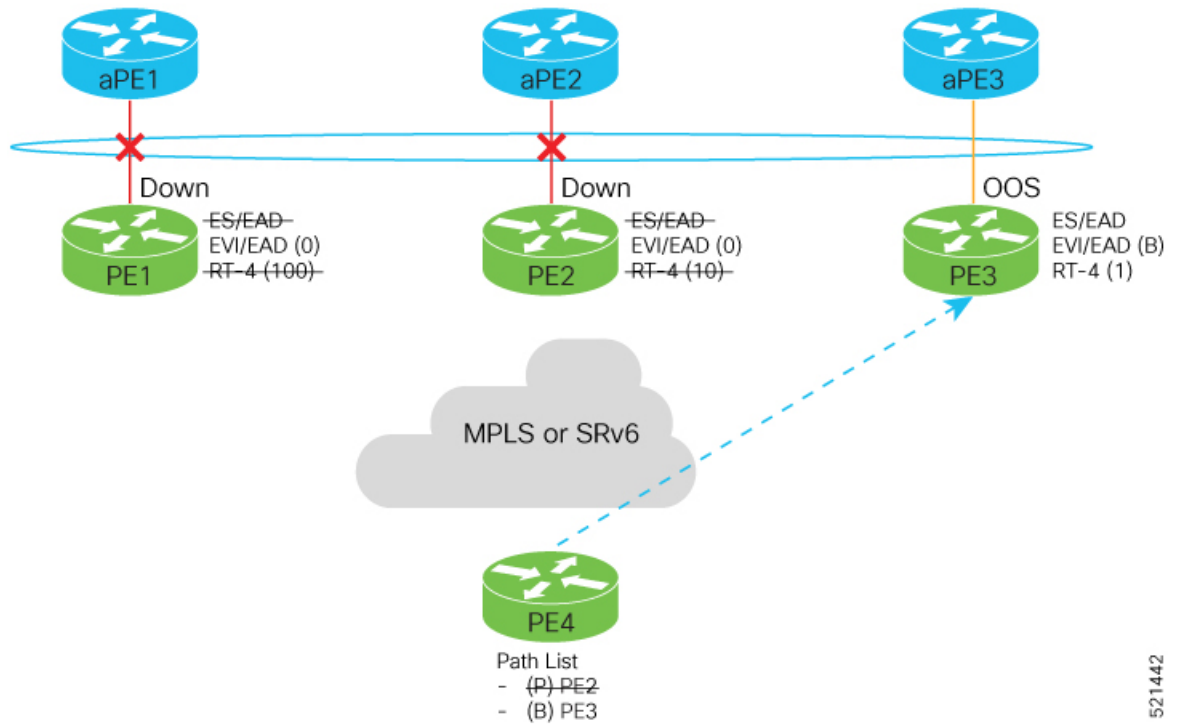
521440



521441

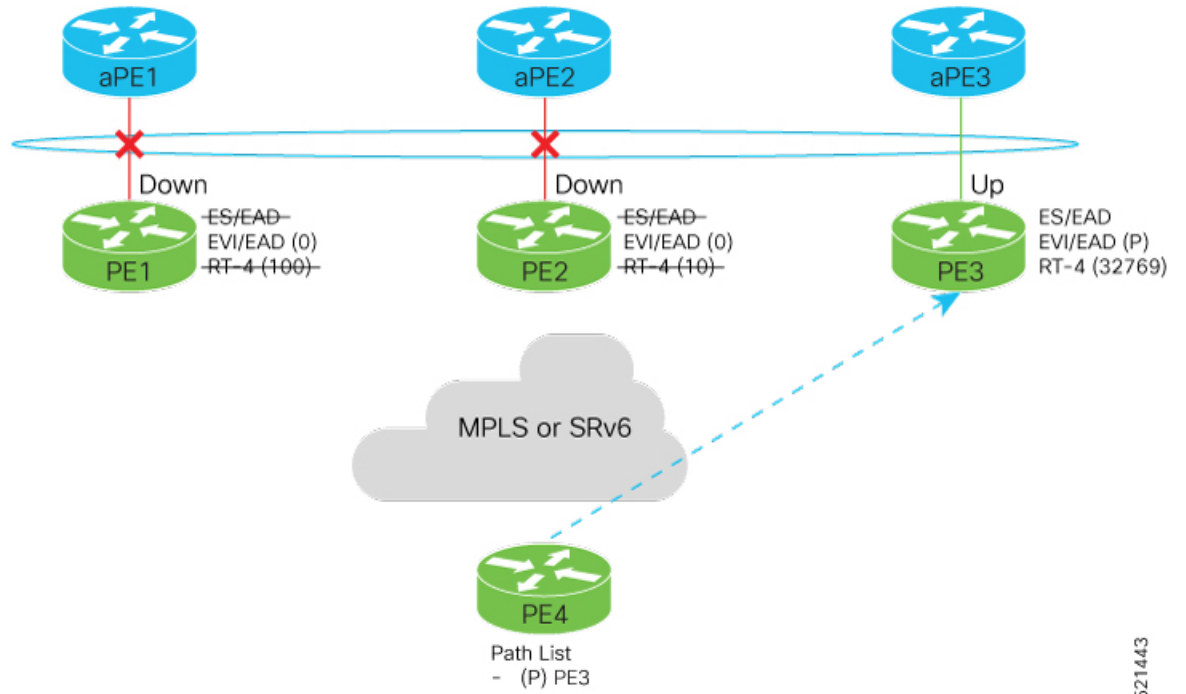
Scenario - 2

Consider a scenario where aPE2-PE2 interface is also down.



521442

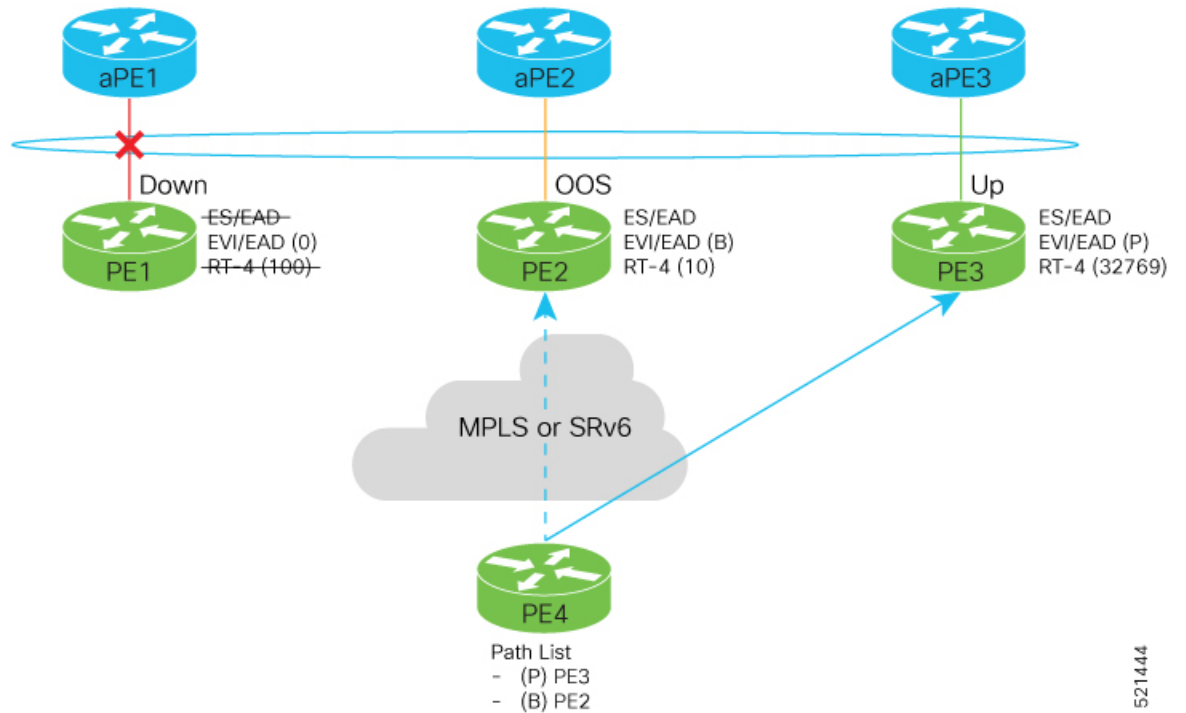
When the aPE2-PE2 interface is also down, the traffic is sent through aPE3-PE3 link. aPE3-PE3 becomes the primary path with a weight of 32769.



521443

Scenario - 3

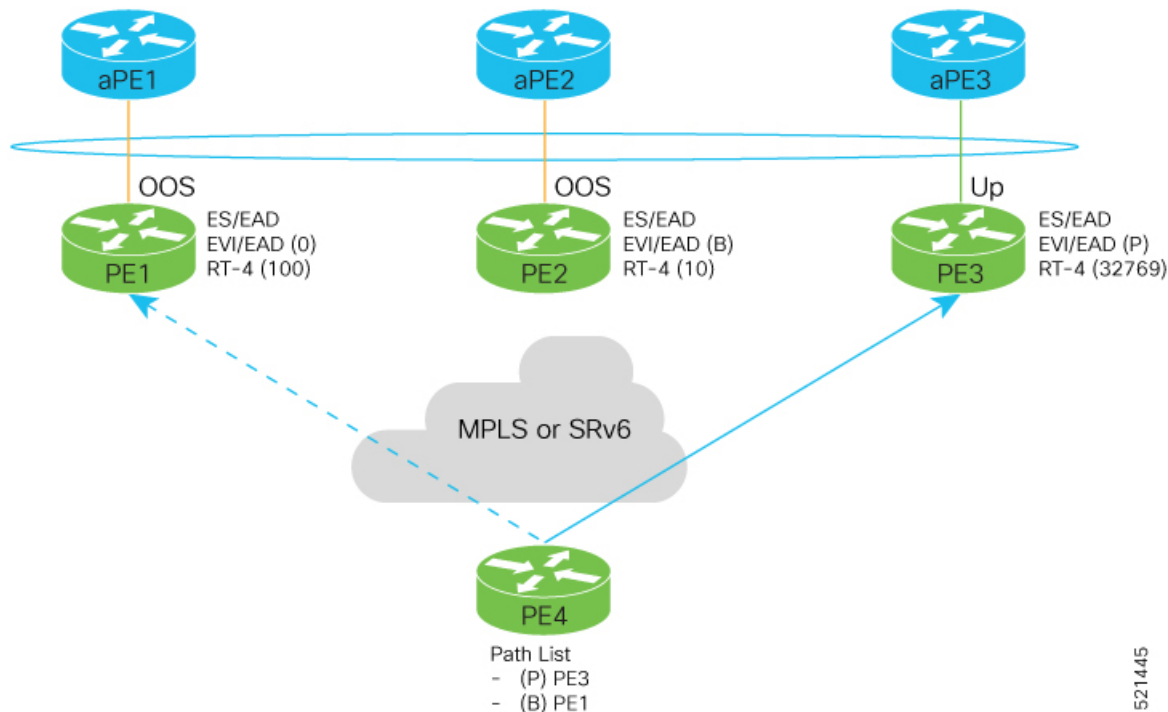
When the aPE2-PE2 interface comes up, the aPE3-PE3 link still remains the primary path. aPE2-PE2 interface becomes the backup path with a weight of 10.



521444

Scenario - 4

When the aPE1-PE1 interface comes up, the aPE3-PE3 link remains the primary path with a weight of 32769. aPE1-PE1 interface becomes the backup path with a weight of 100. The aPE2-PE2 interface becomes NDF with a weight of 10.



521445

CFM Support for EVPN

Ethernet Connectivity Fault Management (CFM) is a service-level OAM protocol that provides tools for monitoring and troubleshooting end-to-end Ethernet services per VLAN. This includes proactive connectivity monitoring, fault verification, and fault isolation. CFM can be deployed in an EVPN network. You can monitor the connections between the nodes using CFM in an EVPN network.

Restrictions

CFM for EVPN is supported with the following restrictions:

- In an active-active multi-homing scenario, when monitoring the connectivity between a multi-homed CE device and the PE devices to which it is connected, CFM can only be used across each individual link between a CE and a PE. Attempts to use CFM on the bundle between CE and PE devices cause sequence number errors and statistical inaccuracies.
- There is a possibility of artefacts in loopback and linktrace results. Either a loopback or linktrace may report multiple results for the same instance, or consecutive instances of a loopback and linktrace between the same two endpoints may produce different results.

For more information about Ethernet Connectivity Fault Management (CFM), refer to the *Configuring Ethernet OAM* chapter in the *Interface and Hardware Component Configuration Guide for Cisco NCS 5500 Series Routers*.

CFM on EVPN ELAN

Connectivity fault management (CFM) is a service-level Operations and Maintenance (OAM) protocol that provides tools for monitoring and troubleshooting end-to-end Ethernet services for each VLAN. This includes proactive connectivity monitoring, fault verification, and fault isolation.

Cisco IOS XR Software Release 6.6.1 introduces CFM support for single-homed EVPN Emulated Local Area Network (ELAN) services. This functionality helps you to monitor the ELAN services of users against their contractual service-level agreements (SLAs), thereby providing high speed Layer 2 and Layer 3 services with high resiliency and less operational complexity to different market segments.

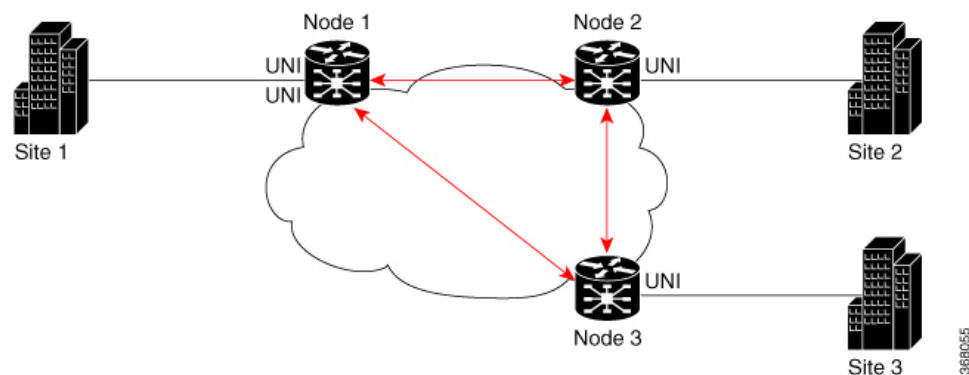
Restrictions for CFM on EVPN ELAN

CFM on EVPN ELAN is subjected to these restrictions:

- Supports only single-homed EVPN ELAN.
- Supports single homing with one AC per PW.
- DOWN MEP on AC interface of EVPN-BD is not supported.
- Does not support loss measurement.
- Does not support Y1731.

Configure CFM on EVPN ELAN

Figure 13: CFM on EVPN ELAN: Full Mesh Topology



Node 1, 2 and 3 in this topology can be Cisco routers.

Configuring CFM on EVPN ELAN involves these main tasks:

- Enabling CFM service continuity check
- Configuring MEP cross-check

- Enabling CFM for the interface

Configuration Example for CFM on EVPN ELAN: Full Mesh Topology

```
/* Enabling CFM continuity check */
Router# ethernet cfm
Router(config-cfm# domain bd-domain level 1 id null
Router(config-cfm-dmn)# service bd-domain bridge group bg-elan bridge-domain bd-elan id
icc-based MC MCMC
Router(config-cfm-dmn-svc)# continuity-check interval 1m
/* Configuring MEP cross-check */
Router(config-cfm-dmn-svc)# mep crosscheck
Router(config-cfm-dmn-svc)# mep-id 1112
Router(config-cfm-dmn-svc)# mep-id 1113
Router(config-cfm-dmn-svc)# commit
```

Repeat the above configurations for node 2 and node 3, with the respective mep-id values. For node 2, configure MEP cross-check with respective mep-id values of node 1 and node 3 (1111 and 1113 respectively, in this example). For node 3, configure MEP cross-check with respective mep-id values of node 1 and node 2 (1111 and 1112 respectively, in this example).

```
/* Enabling CFM on the interface */
Router(config)# interface GigabitEthernet 0/0/0/2.100 l2transport
Router(config-subif)# description bg-elan
Router(config-subif)# encapsulation dot1q 100
Router(config-subif)# rewrite ingress tag pop 1 symmetric
Router(config-subif)# mtu 9100
Router(config-subif)# ethernet cfm
Router(config-if-cfm)# mep domain bd-domain service bd-service mep-id 1111
Router(config-if-cfm-mep)# commit
```

You must repeat the above configurations for node 2 and node 3, with the respective *mep-id* values (that is, 1112 for node 2 and 1113 for node 3, in this example).

Running Configuration for CFM on EVPN ELAN: Full Mesh Topology

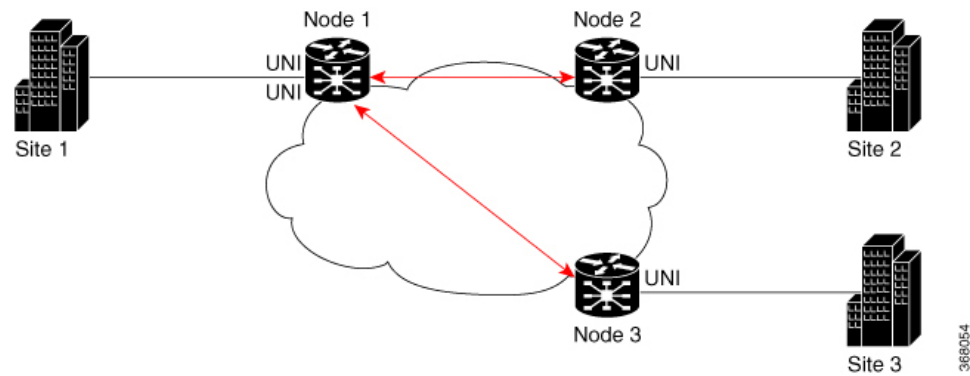
This sections shows the running configuration on node 1.

```
ethernet cfm
 domain bd-domain level 1 id null
  service bd-domain bridge group bg-elan bridge-domain bd-elan id icc-based MC MCMC
  continuity-check interval 1m
  mep crosscheck
  mep-id 1112
  mep-id 1113
  !
!
!

interface GigabitEthernet0/0/0/2.100 l2transport
 description bg-elan
 encapsulation dot1q 100
 rewrite ingress tag pop 1 symmetric
 mtu 9100
 ethernet cfm
  mep domain bd-domain service bd-service mep-id 1111
```

!

Figure 14: CFM on EVPN ELAN: Hub and Spoke Topology



Configuration Example for CFM on EVPN ELAN: Hub and Spoke Topology

The CFM configuration for the hub and spoke topology remains the same as that of full mesh topology mentioned above, except for these additional steps for SLA profile configuration to be done under the interface.

```
/* 1112 and 1113 in this example, are the mep-id values of node 2 and node 3 */
Router(config)#interface gigabitEthernet 0/0/0/2.100 l2transport
Router(config-subif)# ethernet cfm
Router(config-if-cfm)# mep domain bd-domain service bd-service mep-id 1111
Router(config-if-cfm-mep)# sla operation profile test-profile1 target mep-id 1112
Router(config-if-cfm-mep)# sla operation profile test-profile2 target mep-id 1112
Router(config-if-cfm-mep)# sla operation profile test-profile1 target mep-id 1113
Router(config-if-cfm-mep)# sla operation profile test-profile2 target mep-id 1113
Router(config-if-cfm-mep)# commit
```

Running Configuration for CFM on EVPN ELAN: Hub and Spoke Topology

This sections shows the running configuration on node 1.

```
interface GigabitEthernet0/0/0/2.100 l2transport
description bg-elan
encapsulation dot1q 100
rewrite ingress tag pop 1 symmetric
mtu 9100
ethernet cfm
mep domain bd-domain service bd-service mep-id 1111
sla operation profile test-profile1 target mep-id 1112
sla operation profile test-profile2 target mep-id 1112
sla operation profile test-profile1 target mep-id 1113
sla operation profile test-profile2 target mep-id 1113
!
```

Related Topics

[CFM on EVPN ELAN, on page 81](#)

Associated Commands

- continuity-check

- ethernet cfm
- mep crosscheck
- mep domain
- sla operation

EVPN Routing Policy

The EVPN Routing Policy feature provides the route policy support for address-family L2VPN EVPN. This feature adds EVPN route filtering capabilities to the routing policy language (RPL). The filtering is based on various EVPN attributes.

A routing policy instructs the router to inspect routes, filter them, and potentially modify their attributes as they are accepted from a peer, advertised to a peer, or redistributed from one routing protocol to another.

This feature enables you to configure route-policies using EVPN network layer reachability information (NLRI) attributes of EVPN route type 1 to 5 in the route-policy match criteria, which provides more granular definition of route-policy. For example, you can specify a route-policy to be applied to only certain EVPN route-types or any combination of EVPN NLRI attributes. This feature provides flexibility in configuring and deploying solutions by enabling route-policy to filter on EVPN NLRI attributes.

To implement this feature, you need to understand the following concepts:

- Routing Policy Language
- Routing Policy Language Structure
- Routing Policy Language Components
- Routing Policy Language Usage
- Policy Definitions
- Parameterization
- Semantics of Policy Application
- Policy Statements
- Attach Points

For information on these concepts, see [Implementing Routing Policy](#).

Currently, this feature is supported only on BGP neighbor "in" and "out" attach points. The route policy can be applied only on inbound or outbound on a BGP neighbor.

EVPN Route Types

The EVPN NLRI has the following different route types:

Route Type 1: Ethernet Auto-Discovery (AD) Route

The Ethernet (AD) routes are advertised on per EVI and per Ethernet Segment Identifier (ESI) basis. These routes are sent per Ethernet segment (ES). They carry the list of EVIs that belong to the ES. The ESI field is set to zero when a CE is single-homed.

An Ethernet A-D route type specific EVPN NLRI consists of the following fields:

Route Type (1 octet)	*
Length (1 octet)	
Route Distinguisher (RD) (8 octets)	*
Ethernet Segment Identifier (10 octets)	*
Ethernet Tag ID (4 octets)	*
MPLS Label (3 octets)	

NLRI Format: Route-type 1:

[Type] [Len] [RD] [ESI] [ETag] [MPLS Label]

Net attributes: [Type] [RD] [ESI] [ETag]

Path attributes: [MPLS Label]

Example

```
route-policy evpn-policy
  if rd in (10.0.0.1:0) [and/or evpn-route-type is 1] [and/or esi in
(0a1.a2a3.a4a5.a6a7.a8a9)] [and/or etag is 4294967295] then
    set ..
  endif
end-policy
!
route-policy evpn-policy
  if rd in (1.0.0.2:0) [and/or evpn-route-type is 1] [and/or esi in
(00a1.a2a3.a4a5.a6a7.a8a9)] [and/or etag is 4294967295] then
    set ..
  endif
end-policy
```

Route Type 2: MAC/IP Advertisement Route

The host's IP and MAC addresses are advertised to the peers within NLRI. The control plane learning of MAC addresses reduces unknown unicast flooding.

A MAC/IP Advertisement Route type specific EVPN NLRI consists of the following fields:

Route Type (1 octet)	*
Length (1 octet)	
RD (8 octets)	*
Ethernet Segment Identifier (10 octets)	
Ethernet Tag ID (4 octets)	*
MAC Address Length (1 octet)	*
MAC Address (6 octets)	*
IP Address Length (1 octet)	*
IP Address (0, 4, or 16 octets)	*
MPLS Label1 (3 octets)	
MPLS Label2 (0 or 3 octets)	

3063156

NLRI Format: Route-type 2:

[Type][Len][RD][ESI][ETag][MAC Addr Len][MAC Addr][IP Addr Len][IP Addr][MPLS Label1][MPLS Label2]

Net attributes: [Type][RD][ETag][MAC Addr Len][MAC Addr][IP Addr Len][IP Addr]

Path attributes: [ESI], [MPLS Label1], [MPLS Label2]

Example

```
route-policy evpn-policy
  if rd in (10.0.0.2:0) [and/or evpn-route-type is 2] [and/or esi in
(0000.0000.0000.0000)] [and/or etag is 0] [and/or macaddress in (0013.aabb.ccdd)]
[and/or destination in (1.2.3.4/32)] then
  set ..
  endif
end-policy
```

Route Type 3: Inclusive Multicast Ethernet Tag Route

This route establishes the connection for broadcast, unknown unicast, and multicast (BUM) traffic from a source PE to a remote PE. This route is advertised on per VLAN and per ESI basis.

An Inclusive Multicast Ethernet Tag route type specific EVPN NLRI consists of the following fields:

Route Type (1 octet)	*
Length (1 octet)	
RD (8 octets)	*
Ethernet Tag ID (4 octets)	*
IP Address Length (1 octet)	*
Originating Router's IP Address (4 or 16 octets)	*

NLRI Format: Route-type 3:

[Type][Len][RD][ETag][IP Addr Len][Originating Router's IP Addr]

Net attributes: [Type][RD][ETag][IP Addr Len][Originating Router's IP Addr]

Example

```
route-policy evpn-policy
  if rd in (10.0.0.1:300) [and/or evpn-route-type is 3] [and/or etag is 0] [and/or
evpn-originator in (10.0.0.1)] then
    set ..
  endif
end-policy
```

Route Type 4: Ethernet Segment Route

Ethernet segment routes enable to connect a CE device to two or PE devices. ES route enables the discovery of connected PE devices that are connected to the same Ethernet segment.

An Ethernet Segment route type specific EVPN NLRI consists of the following fields:

```

+-----+
|Route Type (1 octet)          |*
+-----+
|Length (1 octet)             |
+-----+
|RD (8 octets)                 |*
+-----+
|Ethernet Segment Identifier (10 octets) |*
+-----+
|IP Address Length (1 octet)   |*
+-----+
|Originating Router's IP Address |*
|(4 or 16 octets)             |
+-----+

```

3/6/2016

NLRI Format: Route-type 4:

[Type][Len][RD][ESI][IP Addr Len][Originating Router's IP Addr]

Net attributes: [Type][RD][ESI][IP Addr Len][Originating Router's IP Addr]

Example

```

route-policy evpn-policy
  if rd in (10.0.0.1:0) [and/or evpn-route-type is 4] [and/or esi in
(00a1.a2a3.a4a5.a6a7.a8a9)] [and/or evpn-originator in (10.0.0.1)] then
    set ..
  endif
end-policy

```

Route Type 5: IP Prefix Route

An IP Prefix Route type specific EVPN NLRI consists of the following fields:

Route Type (1 octet)	*
Length (1 octet)	
RD (8 octets)	*
Ethernet Segment Identifier (10 octets)	
Ethernet Tag ID (4 octets)	*
IP Address Length (1 octet)	*
IP Address (4 or 16 octets)	*
GW IP Address (4 or 16 octets)	
MPLS Label (3 octets)	

NLRI Format: Route-type 5:

[Type][Len][RD][ESI][ETag][IP Addr Len][IP Addr][GW IP Addr][Label]

Net attributes: [Type][RD][ETag][IP Addr Len][IP Addr]

Path attributes: [ESI], [GW IP Addr], [Label]

Example

```
route-policy evpn-policy
  if rd in (30.30.30.30:1) [and/or evpn-route-type is 5] [and/or esi in
(0000.0000.0000.0000.0000)] [and/or etag is 0] [and/or destination in (12.2.0.0/16)] [and/or
evpn-gateway in (0.0.0.0)] then
    set ..
  endif
end-policy
```

EVPN RPL Attribute

Route Distinguisher

A Route Distinguisher (rd) attribute consists of eight octets. An rd can be specified for each of the EVPN route types. This attribute is not mandatory in route-policy.

Example

```
rd in (1.2.3.4:0)
```

EVPN Route Type

EVPN route type attribute consists of one octet. This specifies the EVPN route type. The EVPN route type attribute is used to identify a specific EVPN NLRI prefix format. It is a net attribute in all EVPN route types.

Example

```
evpn-route-type is 3
```

The following are the various EVPN route types that can be used:

```
1 - ethernet-ad
2 - mac-advertisement
3 - inclusive-multicast
4 - ethernet-segment
5 - ip-advertisement
```

IP Prefix

An IP prefix attribute holds IPv4 or IPv6 prefix match specification, each of which has four parts: an address, a mask length, a minimum matching length, and a maximum matching length. The address is required, but the other three parts are optional. When IP prefix is specified in EVPN route type 2, it represents either a IPv4 or IPv6 host IP Address (/32 or /128). When IP prefix is specified in EVPN route type 5, it represents either IPv4 or IPv6 subnet. It is a net attribute in EVPN route type 2 and 5.

Example

```
destination in (128.47.10.2/32)
destination in (128.47.0.0/16)
destination in (128:47::1/128)
destination in (128:47::0/112)
```

esi

An Ethernet Segment Identifier (ESI) attribute consists of 10 octets. It is a net attribute in EVPN route type 1 and 4, and a path attribute in EVPN route type 2 and 5.

Example

```
esi in (ffff.ffff.ffff.ffff.fff0)
```

etag

An Ethernet tag attribute consists of four octets. An Ethernet tag identifies a particular broadcast domain, for example, a VLAN. An EVPN instance consists of one or more broadcast domains. It is a net attribute in EVPN route type 1, 2, 3 and 5.

Example

```
etag in (10000)
```

mac

The mac attribute consists of six octets. This attribute is a net attribute in EVPN route type 2.

Example

```
mac in (0206.acb1.e806)
```

evpn-originator

The evpn-originator attribute specifies the originating router's IP address (4 or 16 octets). This is a net attribute in EVPN route type 3 and 4.

Example

```
evpn-originator in (1.2.3.4)
```

evpn-gateway

The evpn-gateway attribute specifies the gateway IP address. The gateway IP address is a 32-bit or 128-bit field (IPv4 or IPv6), and encodes an overlay next-hop for the IP prefixes. The gateway IP address field can be zero if it is not used as an overlay next-hop. This is a path attribute in EVPN route type 5.

Example

```
evpn-gateway in (1.2.3.4)
```

EVPN RPL Attribute Set

In this context, the term set is used in its mathematical sense to mean an unordered collection of unique elements. The policy language provides sets as a container for groups of values for matching purposes. Sets are used in conditional expressions. The elements of the set are separated by commas. Null (empty) sets are allowed.

prefix-set

A prefix-set holds IPv4 or IPv6 prefix match specifications, each of which has four parts: an address, a mask length, a minimum matching length, and a maximum matching length. The address is required, but the other three parts are optional. The prefix-set specifies one or more IP prefixes.

Example

```
prefix-set ip_prefix_set
14.2.0.0/16,
54.0.0.0/16,
12.12.12.0/24,
50:50::1:0/112
end-set
```

mac-set

The mac-set specifies one or more MAC addresses.

Example

```
mac-set mac_address_set
1234.2345.6789,
2345.3456.7890
end-set
```

esi-set

The esi-set specifies one or more ESI's.

Example

```
esi-set evpn_esi_set
1234.2345.3456.4567.5678,
1234.2345.3456.4567.5670
end-set
```

etag-set

The etag-set specifies one or more Ethernet tags.

Example

```
etag-set evpn_etag_set
10000,
20000
end-set
```

Configure EVPN RPL Feature

The following section describe how to configure mac-set, esi-set, evpn-gateway, and evpn-originator.

```
/* Configuring a mac-set and referring it in a route-policy (Attach point - neighbor-in) */
Router# configure
Router(config)# mac-set demo_mac_set
Router(config-mac)# 1234.ffff.aaa3,
Router(config-mac)# 2323.4444.ffff
Router(config-mac)# end-set
Router(config)# !
Router(config)# route-policy policy_use_pass_mac_set
Router(config-rpl)# if mac in demo_mac_set then
Router(config-rpl-if)# set med 200
Router(config-rpl-if)# else
Router(config-rpl-else)# set med 1000
Router(config-rpl-else)# endif
Router(config-rpl)# end-policy
Router(config)# commit
```

```

Router(config)# router bgp 100
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af) # !
Router(config-bgp-af) # neighbor 10.0.0.10
Router(config-bgp-nbr) # remote-as 8
Router(config-bgp-nbr) # address-family l2vpn evpn
Router(config-bgp-nbr-af) # route-policy policy_use_pass_mac_set in
Router(config-bgp-nbr-af) # commit

/* Configuring a esi-set and referring it in a route-policy (Attach point - neighbor-in) */
Router# configure
Router(config)# esi-set demo_esi
Router(config-esi) # ad34.1233.1222.ffff.44ff,
Router(config-esi) # ad34.1233.1222.ffff.6666
Router(config-esi) # end-set
Router(config) # !
Router(config) # route-policy use_esi
Router(config-rpl) # if esi in demo_esi then
Router(config-rpl-if) # set local-preference 100
Router(config-rpl-if) # else
Router(config-rpl-else) # set local-preference 300
Router(config-rpl-else) # endif
Router(config-rpl) # end-policy
Router(config) # commit

/* Configuring evpn-gateway/evpn-originator in a route-policy (Attach point - neighbor-in
and out) */
Router# configure
Router(config) # route-policy gateway_demo
Router(config-rpl) # if evpn-gateway in (10.0.0.0/32) then
Router(config-rpl-if) # pass
Router(config-rpl-if) # endif
Router(config-rpl) # end-policy
Router(config) # commit
Router(config) # route-policy originator_demo
Router(config-rpl) # if evpn-originator in (10.0.0.1/32) then
Router(config-rpl-if) # set local-preference 100
Router(config-rpl-if) # else
Router(config-rpl-else) # set med 200
Router(config-rpl-else) # endif
Router(config-rpl) # end-policy
Router(config) # commit
Router(config) # router bgp 100
Router(config-bgp) # address-family ipv4 unicast
Router(config-bgp-af) # !
Router(config-bgp-af) # neighbor 10.0.0.10
Router(config-bgp-nbr) # remote-as 8
Router(config-bgp-nbr) # address-family ipv4 unicast
Router(config-bgp-nbr-af) # route-policy gateway_demo in
Router(config-bgp-nbr-af) # route-policy originator_demo out
Router(config-bgp-nbr-af) # commit

```

Running Configuration

```

/* Configuring a mac-set and referring it in a route-policy (Attach point - neighbor-in) */
mac-set demo_mac_set
  1234.ffff.aaa3,
  2323.4444.ffff
end-set
!
route-policy policy_use_pass_mac_set

```

```

    if mac in demo_mac_set then
        set med 200
    else
        set med 1000
    endif
end-policy
!
router bgp 100
  address-family l2vpn evpn
  !
  neighbor 10.0.0.10
    remote-as 8
    address-family l2vpn evpn
    route-policy policy_use_pass_mac_set in
  !
  !
end

/* Configuring a esi-set and referring it in a route-policy (Attach point - neighbor-in) */
Wed Oct 26 11:52:23.720 IST
esi-set demo_esi
  ad34.1233.1222.ffff.44ff,
  ad34.1233.1222.ffff.6666
end-set
!
route-policy use_esi
  if esi in demo_esi then
    set local-preference 100
  else
    set local-preference 300
  endif
end-policy

```

EVPN Route Policy Examples

```

route-policy ex_2
  if rd in (2.2.18.2:1004) and evpn-route-type is 1 then
    drop
  elseif rd in (2.2.18.2:1009) and evpn-route-type is 1 then
    drop
  else
    pass
  endif
end-policy
!
route-policy ex_3
  if evpn-route-type is 5 then
    set extcommunity bandwidth (100:9999)
  else
    pass
  endif
end-policy
!
route-policy samp
end-policy
!
route-policy samp1
  if rd in (30.0.101.2:0) then
    pass
  endif
end-policy

```

```
!
route-policy samp2
  if rd in (30.0.101.2:0, 1:1) then
    pass
  endif
end-policy
!
route-policy samp3
  if rd in (*:*) then
    pass
  endif
end-policy
!
route-policy samp4
  if rd in (30.0.101.2:*) then
    pass
  endif
end-policy
!
route-policy samp5
  if evpn-route-type is 1 then
    pass
  endif
end-policy
!
route-policy samp6
  if evpn-route-type is 2 or evpn-route-type is 5 then
    pass
  endif
end-policy
!
route-policy samp7
  if evpn-route-type is 4 or evpn-route-type is 3 then
    pass
  endif
end-policy
!
route-policy samp8
  if evpn-route-type is 1 or evpn-route-type is 2 or evpn-route-type is 3 then
    pass
  endif
end-policy
!
route-policy samp9
  if evpn-route-type is 1 or evpn-route-type is 2 or evpn-route-type is 3 or evpn-route-type
  is 4 then
    pass
  endif
end-policy
!
route-policy test1
  if evpn-route-type is 2 then
    set next-hop 10.2.3.4
  else
    pass
  endif
end-policy
!
route-policy test2
  if evpn-route-type is 2 then
    set next-hop 10.10.10.10
  else
    drop
  endif
end-policy
```

```
end-policy
!
route-policy test3
  if evpn-route-type is 1 then
    set tag 9988
  else
    pass
  endif
end-policy
!
route-policy samp21
  if mac in (6000.6000.6000) then
    pass
  endif
end-policy
!
route-policy samp22
  if extcommunity rt matches-any (100:1001) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp23
  if evpn-route-type is 1 and esi in (aaaa.bbbb.cccc.dddd.eeee) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp24
  if evpn-route-type is 5 and extcommunity rt matches-any (100:1001) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp25
  if evpn-route-type is 2 and esi in (1234.1234.1234.1234.1236) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp26
  if etag in (20000) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp27
  if destination in (99.99.99.1) and etag in (20000) then
    pass
  else
    drop
  endif
end-policy
!
```



```
route-policy samp31
  if evpn-route-type is 1 or evpn-route-type is 2 or evpn-route-type is 3 or evpn-route-type
  is 4 or evpn-route-type is 5 then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp33
  if esi in evpn_esi_set1 then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp34
  if destination in (90:1:1::9/128) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp35
  if destination in evpn_prefix_set1 then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp36
  if evpn-route-type is 3 and evpn-originator in (80:1:1::3) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp37
  if evpn-gateway in (10:10::10) then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp38
  if mac in evpn_mac_set1 then
    pass
  else
    drop
  endif
end-policy
!
route-policy samp39
  if mac in (6000.6000.6002) then
    pass
  else
    drop
  endif
end-policy
```

```
!  
route-policy samp41  
  if evpn-gateway in (10.10.10.10, 10:10::10) then  
    pass  
  else  
    drop  
  endif  
end-policy  
!  
route-policy samp42  
  if evpn-originator in (24.162.160.1/32, 70:1:1::1/128) then  
    pass  
  else  
    drop  
  endif  
end-policy  
!  
route-policy example  
  if rd in (62300:1903) and evpn-route-type is 1 then  
    drop  
  elseif rd in (62300:19032) and evpn-route-type is 1 then  
    drop  
  else  
    pass  
  endif  
end-policy  
!  
route-policy samp100  
  if evpn-route-type is 4 or evpn-route-type is 5 then  
    drop  
  else  
    pass  
  endif  
end-policy  
!  
route-policy samp101  
  if evpn-route-type is 4 then  
    drop  
  else  
    pass  
  endif  
end-policy  
!  
route-policy samp102  
  if evpn-route-type is 4 then  
    drop  
  elseif evpn-route-type is 5 then  
    drop  
  else  
    pass  
  endif  
end-policy  
!  
route-policy samp103  
  if evpn-route-type is 2 and destination in evpn_prefix_set1 then  
    drop  
  else  
    pass  
  endif  
end-policy  
!  
route-policy samp104  
  if evpn-route-type is 1 and etag in evpn_etag_set1 then  
    drop
```

```
elseif evpn-route-type is 2 and mac in evpn_mac_set1 then
    drop
elseif evpn-route-type is 5 and esi in evpn_esi_set1 then
    drop
else
    pass
endif
end-policy
!
```

